AI in Computer Games: Generating Interesting Interactive Opponents by the use of Evolutionary Computation

Georgios N. Yannakakis

Doctor of Philosophy

Institute of Perception, Action and Behaviour
School of Informatics
University of Edinburgh

2005

Abstract

Which features of a computer game contribute to the player's enjoyment of it? How can we automatically generate interesting and satisfying playing experiences for a given game? These are the two key questions addressed in this dissertation.

Player satisfaction in computer games depends on a variety of factors; here the focus is on the contribution of the behaviour and strategy of game opponents in predator/prey games. A quantitative metric of the 'interestingness' of opponent behaviours is defined based on qualitative considerations of what is enjoyable in such games, and a mathematical formulation grounded in observable data is derived. Using this metric, neural-network opponent controllers are evolved for dynamic game environments where limited inter-agent communication is used to drive spatial coordination of opponent teams.

Given the complexity of the predator task, cooperative team behaviours are investigated. Initial candidates are generated using off-line learning procedures operating on minimal neural controllers with the aim of maximising opponent performance. These example controllers are then adapted using on-line (i.e. during play) learning techniques to yield opponents that provide games of high interest. The on-line learning methodology is evaluated using two dissimilar predator/prey games with a number of different computer player strategies. It exhibits generality across the two game test-beds and robustness to changes of player, initial opponent controller selected, and complexity of the game field.

The interest metric is also evaluated by comparison with human judgement of game satisfaction in an experimental survey. A statistically significant number of players were asked to rank game experiences with a test-bed game using perceived interestingness and their ranking was compared with that of the proposed interest metric. The results show that the interest metric is consistent with human judgement of game satisfaction.

Finally, the generality, limitations and potential of the proposed methodology and techniques are discussed, and other factors affecting the player's satisfaction, such as the player's own strategy, are briefly considered. Future directions building on the work described herein are presented and discussed.

Acknowledgements

This is apparently the only section of this dissertation that I write without having to have it reviewed and I am so glad about it.

John Hallam, I owe you a big thank you for all your support, your supervision, but most of all, for the exciting conversations we had. Our meetings and discussions in Edinburgh, Odense and Crete have been more than inspiring and motivating. Thanks for helping me see research from your perspective.

I am also very grateful to my second supervisor John Levine. John's help was very significant in the beginning of my studies and his careful research guidance in the first two years has led to this thesis' successful completion. I would also like to thank Craig Robertson. A single meeting with him, when he was in IPAB, was enough to convince me to move my research towards games. I am very happy that I did. Thanks Craig!

Yiorgos Sideris has highly contributed to the development of the Pac-Man interface and saved me loads of time by providing all the technical support I needed at that time. Jeong Kun Park was the first to come up with the idea of the Dead End game concept as part of his MSc degree in Edinburgh. Thank you both for your contributions.

Thanks also to all people from the Maersk Institute in Odense for hosting me there and especially to *Adaptronics* people for their support and help to integrate as much as I could to the Danish way of living.

Loads of thanks to the people that reviewed and commented parts of this script: Markos Papageorgiou, Efstathia Alexiou and Kasper Støy. I am also very grateful to all those that participated to my experiments.

Last, but definitely not least, I would like to thank my parents and my sister for their constant support and love. Nothing of all this would be possible without them.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Georgios N. Yannakakis*)

To Dimitris, Piyi, Yiorgos and Antonia; my grandparents

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Computer games constitute a major branch of the entertainment industry nowadays. The financial and research potential of making games more appealing (or else more interesting) is more than impressive. Artificial intelligence (AI) (Russell and Norvig, 1995) is one of the fields that will benefit from humans' constant demand for more intelligent and interesting games. Furthermore, the continuous increase of computing power can bring expensive, until recently, AI techniques back to life in a computer game application. Machine learning techniques are able to produce characters with intelligent capabilities. On that basis, interactive and cooperative characters can generate more realism to games and satisfaction to the player.

1.1 Motivation

The computer game industry has grown rapidly over the last decade. In 2004, sales of computer games in the U.S. reached a total of 7.6 billion U.S. dollars, while in 1996 the respective amount was 3.0 billion U.S. dollars according to the latest reports of the Entertainment Software Association (2004). Currently computer games is the biggest sector of the entertainment industry in the U.S. and among the wealthiest industry sectors in the U.K. Computer games' advantage in contrast with other forms of entertainment is their interactivity in combination with their relatively low price. Note also that the constant broadening of the age and social target groups has also contributed to their popularity.

Computer games are based on various concepts which determine their core and classify

them in genres. This form of digital entertainment is embedded in a virtual simulated world that the player interacts with. Such virtual worlds have seen major audiovisual improvements over the last twenty years; from abstract two-dimensional surfaces to complex realistic worlds. This graphical realism has been the key focus of game developers and has contributed to the increasing popularity of games. Consequently, realistic game worlds have reached a point nowadays where there can only be slow and minor further enhancements in that direction (Champandard, 2004).

While game development has been concentrated primarily on the graphical representation of the game worlds, minor focus has been given on the non-player characters' (NPCs') behavior. Simple scripted rules and finite-state or fuzzy-state machines are still used to control NPCs in the majority of games (Woodcock, 2001). The increasing number of multi-player online games (among others) is an indication that humans seek more intelligent opponents and richer interactivity. Advanced artificial intelligence techniques are able to improve gaming experience by generating intelligent interactive characters and furthermore cover this human demand (Funge, 2004). Moreover, computational power may bring expensive innovative AI techniques such as machine learning to meet a game application in the near future.

Intelligent interactive opponents can provide more enjoyment to a vast gaming community of constant demand for more realistic, challenging and meaningful entertainment (Fogel *et al.*, 2004). However, given the current state-of-the-art in AI in games, it is unclear which features of any game contribute to the satisfaction of its players, and thus it is also uncertain how to develop enjoyable games. Because of this lack of knowledge, most commercial and academic research in this area is fundamentally incomplete.

The challenges we consider in this dissertation are to provide qualitative and quantitative means for distinguishing a game's enjoyment value and to develop efficient tools to automatically generate entertainment for the player. In that sense, investigation of the factors/criteria that map to real-time enjoyment for the player as well as the mechanisms that are capable of generating highly entertaining games constitute our primary aims. To achieve our goals, we consider specific prerequisites (presented in the following section) that determine our milestones towards enhancing this form of digital entertainment.

1.2 Questions and Objectives

Given the research motivations discussed in the previous section, there are specific issues that need to be addressed towards our objectives. The research questions that will be answered in this thesis are as follows.

1. Which are the features/criteria that collectively determine enjoyment in computer games.
2. How to quantitatively measure the player's satisfaction (i.e. interest) in real-time.
3. How to increase a game's low interest and/or how to maintain a highly interesting game.
4. How the player may affect his/her entertainment while playing.

The two primary objectives of this thesis as well as the sub-goals that the aforementioned questions generate are as follows.

- Establish an efficient interest metric for computer games and demonstrate its generality. In this dissertation we will experiment with predator/prey games as a test bed.
- Enhance entertainment of computer players based on this metric. This is achieved through evolutionary learning of neural-controlled opponents in real-time (while playing).

Our hypothesis is that opponents with minimal controllers that demonstrate emergent cooperative behaviors are able to enhance entertainment in a computer game. Moreover, if their intra-communication is based on indirect (implicit), non-global (i.e. partial) and 'passive' information (i.e. information that does not alter the state of the game world itself), opponents are able to demonstrate a more robust behavior and generality over the complexity of the game environment in contrary to opponents with global sensing of their environment. Minimal knowledge of the game world, that is able to generate desired behaviors, is one of the features of the opponents used in this thesis.

Therefore, we investigate off-line learning procedures able to generate cooperative action in such game environments and furthermore, explore techniques to minimize the opponents' neural-controllers for optimal real-time performance. Given the cooperative action and the minimal controller structure of the opponents, we have designed

a successful evolutionary learning procedure applied in games in real-time (see Section 1.4 for the challenges of learning in real-time).

1.3 Game Properties

Cooperative behaviors within systems (environments, artificial game worlds) of multiple agents is a prominent area of research. Designing agents for such systems could be a repetitive and tedious procedure. This task becomes even more difficult when the multi-agent environment investigated is highly dynamic and non-deterministic. Additional complication is present when agents' communication is implicit and partial. Thus, when designing controllers for autonomous simulated agents for such environments, there is little guidance on how complex the controller must be for the agents to achieve good performance in particular tasks. Furthermore, when such a performance is to emerge via a learning mechanism, there is little knowledge about the mechanism's design and complexity.

The test-bed game environments used for our experiments are designed according to the following features:

1. Two dimensional.
2. Multi-agent (i.e. multiple opponents) for studying emergent cooperation.
3. Agent communication is based on implicit, partial and passive information. We make this choice by following principles of the *animat* approach (Meyer and Guillot, 1994) which provides the ground for realistic NPC behaviors in computer games (Champandard, 2004).
4. Agents' tasks investigated are limited to spatial coordination.

Our first objective in developing such (game) worlds is to investigate the potential emergence of cooperative complex opponent behaviors amongst the agents given their type of communication and specific tasks they have to achieve. Subsequently, we investigate how these behaviors may impact the player's perceived entertainment.

A set of games that collectively embodies all the above-mentioned environment features is the predator/prey genre. We choose predator/prey games as the initial genre of our game research since, given our aims, they provide us with unique properties.

In such games we can deliberately abstract the environment and concentrate on the characters' behavior. The examined behavior is cooperative since cooperation is a pre-requisite for effective hunting behaviors. Furthermore, we are able to easily control a learning process through on-line interaction. In other words, predator/prey games offer a well-suited arena for initial steps in studying cooperative behaviors generated by interactive on-line learning mechanisms. Other genres of game (e.g. first person shooters) offer similar properties; however predator/prey games are chosen for their simplicity as far as their development and design are concerned.

Note also that our experiments are held in a personal computer rather than an online server. Thus, both the game real-time graphical presentation and all computational processes that run on its background are hosted in a single CPU. This selection bounds the available computational power and generates challenges for the fast adaptation of any on-line learning opponents. However, it is a realistic choice since the majority of computer games released run on a single CPU and this should be the framework of the application of any advanced AI mechanism (Woodcock, 2001).

1.4 Learning in Real-Time Challenges

We can distinguish the challenges we will come across in this dissertation into the ones generated by the game design per se and the ones generated by learning (off-line or on-line) in computer games. The former challenges will be extensively presented after each game's description, off-line learning challenges will be discussed in Chapter 4, whereas the on-line learning challenges are introduced in this section. We make this distinction here since the difficulties arising from learning in real-time will be faced globally throughout the thesis.

The drawbacks we have to overcome when designing a learning (neuro-evolution) mechanism for real-time opponent adaptation are as follows:

- **Real-time performance**. An on-line learning approach should perform fast in real-time since only a single CPU is available for the majority of computer games (Woodcock, 2001). Note also, that most commercial computer games use over 90% of the CPU for their graphics engines only.
- **Realism**. On-line neuro-evolution provides the potential for adaptive opponents but it may also generate unrealistic opponent behaviors. In most computer games

the player interacts with a very small number of opponents in the majority of gameplay instances. Unrealistic behaviors are, therefore, easily noticeable and may lead to low entertainment for the player.

- **Fast adaptation**. A successful on-line learning approach should adapt quickly to changes in the player's strategy. Otherwise, the game becomes boring. This constitutes a big challenge for the design of the on-line learning mechanism given the computational power resources, the realistic behavior condition presented before and the small number of opponents that normally interact with the player in computer games.

The predator/prey games used in this dissertation face all aforementioned challenges of on-line learning design since experiments are held in a single 1GHz CPU and the number of opponents is less than or equal to five.

1.5 Summary

Mainly motivated by the current lack of a qualitative and quantitative entertainment formulation of computer games and the procedures to generate it, this dissertation covers the following issues: It presents the features — extracted primarily from the opponent behavior — that make a predator/prey game appealing; provides the qualitative and quantitative means for measuring player entertainment in real-time and introduces a successful methodology for obtaining games of high satisfaction. This methodology is based on on-line (during play) learning opponents who embed minimal controllers and demonstrate cooperative action.

1.6 Summary of Thesis

The thesis is organized into chapters as follows.

Chapter 2 introduces the criteria that make predator/prey games interesting and quantifies them in a formula. The assumptions over which this metric is formulated and furthermore the metric's potential generality are also discussed.

Chapter 3 provides an outline of the computer games' state of industry and the state-of-the-art of AI in games. Extensive literature review of the tools and the methodology

used are also presented in this chapter.

Chapter 4 describes the methodological approaches followed in this thesis. In particular this includes: emergence of cooperative opponent behaviors through off-line learning mechanisms; minimization of opponents' controllers and game's interest enhancement through on-line learning procedures. This chapter also introduces a prototype test-bed for the study of emergent cooperation in simulated virtual environments.

Chapter 5 introduces the predator/prey game called 'Dead End' and demonstrates experiments of our approach in this test-bed.

Chapter 6 demonstrates the generality of the method over the predator/prey genre by introducing experiments on a modified version of the well-known Pac-Man game.

Chapter 7 presents a survey based on human subjects, which attempts to draw the correlation between human notion of interestingness and the interest metric proposed and to test the on-line evolutionary learning mechanism under real gaming conditions.

Chapter 8 investigates the player's impact on the real-time interest of the game. Experiments with player modeling techniques provide some first insights for this form of game-player interaction. In addition, future research steps beyond the limits of this dissertation are discussed.

Chapter 9 summarizes the thesis' main achievements and contributions and discusses the proposed methodology's current limitations. Moreover, potential solutions that might embrace these drawbacks are presented.

Chapter 2

Interest

According to the thesis' objectives described in Chapter 1, the first step towards generating enjoyable computer games is to empirically discuss the criteria or features of games that collectively define enjoyment (or else interest) in computer games. Our application area is the genre of predator/prey games because of the distinctive features and advantages they offer (see Chapter 1).

In this chapter we discuss entertainment metrics in computer games, previous research endeavors in the area as well as their importance regarding the area of AI in games. Subsequently, we introduce a mathematical presentation of interest in predator/prey games based on criteria that define entertainment in such games. Finally, we outline the essential parameter configurations for the proposed metric and present the assumptions on which this metric is built.

2.1 Entertainment Metrics

Research in the AI in computer games field is based on several empirical assumptions about human cognition and human-machine interaction. The primary hypothesis is that by generating human-like opponents (Freed *et al.*, 2000), computer games become more appealing and enjoyable. While there are indications to support such a hypothesis (e.g. the vast number of multi-player on-line games played daily on the web) and recent research endeavors to investigate the correlation between believability of NPCs and satisfaction of the player (Taatgen *et al.*, 2003), there has been no evidence that a specific opponent behavior generates more or less interesting games.

Iida's work on measures of entertainment in board games was the first attempt in this area. He introduced a general metric of entertainment for variants of chess games depending on average game length and possible moves (Iida *et al.*, 2003). On that basis, some endeavors towards the criteria that collectively make simple online games appealing are discussed in (Crispini, 2003). The human survey-based outcome of that work presents challenge, diversity and unpredictability as primary criteria for enjoyable opponent behaviors.

2.2 Interest in Predator/Prey Computer Games

As noted in Chapter 1, predator/prey games will be our test-bed genre for the investigation of enjoyable games. More specifically, in the games studied, the prey is controlled by the player and the predators are the computer-controlled opponents. In order to find an objective measure of interest in such games we first need to empirically define the criteria that make such a game interesting. Then, second, we need to quantify and combine all these criteria in a mathematical formula[1]. Subsequently, a test-bed game should be tested by human players to have this formulation of interest cross-validated against the interest the game produces in real conditions (see Chapter 7).

To simplify this procedure we will ignore the graphics' and the sound effects' contributions to the interest of the game and we will concentrate on the opponents' behaviors. That is because, we believe, the computer-guided opponent character contributes the vast majority of features that make a computer game interesting. The player, however, may contribute to its entertainment through its interaction with the opponents of the game and, therefore, it is implicitly included in the interest formulation presented here — see also Chapter 8 for studies on the player's impact on his/her entertainment.

2.2.1 Criteria

By observing the opponents' behavior of various predator/prey games we attempted to empirically extract the features that may generate entertainment for the player. These features were experimentally cross-validated against various opponents of different strategies and redefined when appropriate. Hence, by being as objective and generic

[1] Parts of this chapter's material have been published in (Yannakakis and Hallam, 2004a; 2004b; 2005d)

as possible, we believe that the criteria that collectively define interest on any predator/prey game are as follows.

1. *When the game is neither too hard nor too easy*. In other words, the game is interesting when predators (opponents) manage to kill the prey (player) sometimes but not always. In that sense, given a specific game structure and a player, highly effective opponent behaviors are not interesting behaviors and *vice versa*.

2. *When there is diversity in opponents' behavior over the games*. That is, when the NPCs are able to find dissimilar ways of hunting and killing the player in each game so that their strategy is less predictable.

3. *When opponents' behavior is aggressive rather than static*. That is, predators that move constantly all over the game world and cover it uniformly. This behavior gives the player the impression of an intelligent strategic opponents' plan which increases the game interest.

2.2.2 Metrics

In order to estimate and quantify each of the three aforementioned interest criteria, we let a group of game opponents — the number of opponents depends on the specific game under examination — play the game N times (each game for a sufficiently large evaluation period of t_{max} simulation steps) and we record the simulation steps t_k taken to kill the player as well as the total number of the opponents' visits v_{ik} at each cell i of the grid game field for each game k. In the case where the game's motion is continuous, a discretization of the field's plane up to the character's size can serve this purpose.

Given these, the quantifications of the three interest criteria proposed above can be presented as follows.

1. Appropriate Level of Challenge. According to the first criterion, an estimate of how interesting the behavior is, is given by T in (2.1).

$$T = [1 - (E\{t_k\}/\max\{t_k\})]^{p_1} \tag{2.1}$$

where $E\{t_k\}$ is the average number of simulation steps taken to kill the prey-player over the N games; $\max\{t_k\}$ is the maximum t_k over the N games — $\max\{t_k\} \leq t_{max}$; p_1 is a weighting parameter.

The T estimate of interest demonstrates that the greater the difference between the average and the maximum number of steps taken to kill the player, the higher the interest of the game. Given (2.1), both easy-killing ('*too easy*') and near-optimal ('*too hard*') behaviors get low interest estimate values (i.e. $E\{t_k\} \simeq \max\{t_k\}$). This metric is also called 'challenge'.

2. Behavior Diversity. The interest estimate for the second criterion is given by S in (2.2).

$$S = (\sigma_{t_k}/\sigma_{max})^{p_2} \tag{2.2}$$

where

$$\sigma_{max} = \frac{1}{2}\sqrt{\frac{N}{(N-1)}}(t_{max} - t_{min}) \tag{2.3}$$

and σ_{t_k} is the standard deviation of t_k over the N games; σ_{max} is an estimate of the maximum value of σ_{t_k}; t_{min} is the minimum number of simulation steps required for predators to kill the prey obtained by playing against some 'well' behaved fixed strategy near-optimal predators ($t_{min} \leq t_k$); p_2 is a weighting parameter.

The S estimate of interest demonstrates that the greater the standard deviation of the steps taken to kill the player over N games, the higher the interest of the behavior. Therefore, by using (2.2) we promote predators that produce high diversity in the time taken to kill the prey.

3. Spatial Diversity. A good measure for quantifying the third interest criterion is through entropy of the predators' cell visits in a game, which quantifies the completeness and uniformity with which the opponents cover the stage. Hence, for each game, the cell visit entropy is calculated and normalized into $[0,1]$ via (2.4).

$$H_n = \left[-\frac{1}{logV_n}\sum_i \frac{v_{in}}{V_n} log\left(\frac{v_{in}}{V_n}\right)\right]^{p_3} \tag{2.4}$$

where V_n is the total number of visits of all visited cells (i.e. $V_n = \sum_i v_{in}$) and p_3 is a weighting parameter.

Given the normalized entropy values H_n for all N games, the interest estimate for the third criterion can be represented by their average value $E\{H_n\}$ over the N games. This implies that the higher the average entropy value, the more interesting the game becomes.

All three criteria metrics are combined linearly (2.5)

$$I = \frac{\gamma T + \delta S + \varepsilon E\{H_n\}}{\gamma + \delta + \varepsilon} \tag{2.5}$$

where I is the interest value of the predator/prey game; γ, δ and ε are criterion weight parameters.

To obtain the I value's confidence intervals we follow the bootstrapping procedure presented in Appendix A. The I value proposed here is a reliable estimate of player entertainment in real-time since it is approved by humans (see Chapter 7).

This thesis presents an innovative and efficient approach to model and quantify entertainment; however, without claiming of this approach being unique. We believe that other successful quantitative metrics for the appropriate level of challenge, the opponents' diversity and the opponents' spatial diversity may be designed and more qualitative criteria may be inserted in the interest formula. For example other metrics for measuring the appropriate level of challenge metric could be used: one could come up with a T metric assuming that the appropriate level of challenge follows a Gaussian distribution over $E\{t_k\}$ and that the interest value of a given game varies depending on how long it is — *very short* ($E\{t_k\} \approx t_{min}$) games tend to be frustrating and *long games* ($E\{t_k\} \approx max\{t_k\}$) tend to be boring. However, very short games are not frequent in the experiments presented in this thesis and, therefore, by varying the weight parameter p_1 in the proposed T metric (see (2.1)) we are able to obtain an adequate level of variation in challenge.

2.3 Metric's Generality

The interest metric introduced in (2.5) can be applied effectively to any predator/prey computer game because it is based on generic features of this category of games. These features include the time required to kill the prey and the predators' entropy throughout the game field. We therefore believe that (2.5) — or a similar measure of the same concepts — constitutes a generic interest approximation of predator/prey computer games. Moreover, given the two first interest criteria previously defined, the approach's generality is expandable to all computer games. Indeed, no player likes any computer game that is too hard or too easy to play and, furthermore, any player would enjoy diversity throughout the play of any game. The third interest criterion is applicable

to games where spatial diversity is important which, apart from predator/prey games, may also include action, strategy and team sports games according to the computer game genre classification of Laird and van Lent (2000).

Evidence demonstrating the interest metric's generality appears in Chapter 5 and Chapter 6 through experiments on dissimilar predator/prey games. Moreover, this metric's potential extensibility beyond the predator/prey genre is discussed in Chapter 9.

2.4 Weighting Parameters

There are six weighting parameters that affect the interest value and are classified in two categories:

- power parameters: p_1, p_2 and p_3 and
- criterion parameters: γ, δ and ε.

In order to obtain values for the weighting parameters p_1, p_2 and p_3 we select empirical values based on each interest criterion.

- For the first interest metric presented in (2.1), p_1 is adjusted so as to give T a greater impact or else a boost when even a slight difference between the maximum and the average life time of the player (i.e. challenge) is noted. In that sense, by selecting values of $p_1 < 1$ we reward quite challenging opponents more than near-optimal killers.
- For the second interest metric presented in (2.2), p_2 is set so as σ_{t_k} has a linear effect on S.
- For the third interest metric presented in (2.4), p_3 is adjusted in order to press for very high H_n values and furthermore to provide a clearer distinction between high and low normalized entropy values. Appropriate p_3 parameter values which serve this purpose are those greater than one.

By taking the above into consideration, we come up with $p_1 = 0.5$, $p_2 = 1$ and $p_3 = 4$. These power parameter values will be fixed and independent of the game test-beds used throughout this thesis.

As far as the weighting parameters γ, δ and ε are concerned, they constitute game parameters (see Section 2.5) which means that they are empirically adjusted according

to each predator/prey game's features.

2.4.1 Sensitivity of the I Value

Sensitivity analysis of the I value in respect to the six weighting parameters is conducted prior to any test-bed game application. This procedure allows for power parameter (p_1, p_2 and p_3) values to be tested and criterion parameter (γ, δ and ε) values to be properly adjusted. Chapter 5 and Chapter 6 present comprehensive sensitivity analysis of the I value when applied in two predator/prey computer games.

2.5 Game Parameters

The parameters that need to be adjusted for the successful application of the interest metric (2.5) in a potential predator/prey game are as follows:

- N is the number of games required for I to be estimated. This number is 50 (except when otherwise noted) throughout the thesis and it depends on the game application and the available computational power. As expected, the larger the number N of games, the better the interest value approximation.
- t_{min} is an estimate of the minimum number of simulation steps required for predators to kill the prey and is normally obtained through several trials of hand-coded near-optimal predators against various player types (hand-coded or human).
- t_{max} is an estimate of the maximum duration of each of the N games and it is also obtained through experimentation. This parameter determines a game period in which the player's (prey's) tasks regarding the specific game are accomplished. Normally, well-behaved handcrafted preys are used to adjust this time period by playing against different types of predators. Expert-skilled human players could be used when the handcrafted design of a well-behaved prey's controller fails due to high complexity of the gameplay.
- γ, δ and ε are the criterion weighting parameters presented in (2.5). As mentioned in Section 2.4 these parameter values are empirically adjusted according to each predator/prey game's features. In particular, these values are set in re-

spect to the emphasis that each game gives to each interest criterion. For instance, the interest of some predator/prey games might be conceptually more dependent on the spatial activity of the opponents than challenge or the diversity of the opponents' behavior and vice versa.

Various baseline opponent strategies are used to confirm the I value grounded in the selected parameters by comparing their observed behaviors. Examples of γ, δ and ε parameter value selection appear in Chapter 5 and Chapter 6.

2.6 Assumptions

The proposed interest metric is based on specific assumptions that are covered in this section. The assumptions drawn at this stage will be followed throughout the thesis and are as follows.

- The interest metric is an estimate of the opponents' overall contribution to the game's interest. The methodology followed is based on a number of games N that the player has to play in order for the interest value to be calculated. The assumption of N games is consistent with human cognition since it appears that human players require a significant number of games (or else playing time) to classify a specific computer game according to their perceived satisfaction.
- The interest value is built on the assumption that players of the game have average-playing skills. By 'average-playing' we only exclude the two following extreme player types:
  1. Players that have never played a specific game before and furthermore do not know the rules and how to play it. These players lose constantly against almost any type of opponent.
  2. Players that have an excellent knowledge of a specific game, can easily predict the opponent behavior and have mastered the game controls. These players can beat even the most efficient opponents designed for this game.

  In both cases, the interest value might not be very well estimated since the challenge criterion T approximates a zero value regardless of the opponent. Thus, both player types produce low interest values due to the low challenge criterion ($T \approx 0$). This appears to be consistent with human notion of interestingness

since we believe that neither extreme player type is conceptually interested in the game.

Even though the player types mentioned above are rare, they constitute a limitation of the methodology that is further discussed in Chapter 9.

- The interest value is dependent primarily on the opponents' behavior and implicitly on the player's behavior through the interaction of the two. Any game feature apart from the 'intelligence' of the game characters is not considered to affect the I value. More specifically, features such as graphics, sound effects, game concept etc. are not included in formula (2.5) and are not covered in this thesis. Our assumption here is that the core of the qualitative characteristics of interestingness can be found in the game characters' behavior. Any other game feature is considered peripheral as far as this thesis is concerned. See a comprehensive discussion of this assumption in Chapter 8 where the interest value dependence on the player is investigated through experiments in a test-bed game.
- A factor that may contribute to enjoyable games is the real-time speed of the game and the reaction time scale of the player. The gradual decrease of the required real-time reaction time is a standard and inexpensive technique used by a set of games (i.e Pac-Man) in order to achieve higher challenge for the player as the game proceeds. However, as game speed is grounded in the game design per se and does not alter the quality of the opponent behavior (as it does for the human player behavior), it is not examined in this dissertation. Note that in the extreme case, an unintelligent opponent may be performing effectively because of the unrealistically fast reaction time required by humans.

2.7 Summary

In this chapter we introduced an efficient method for obtaining a quantitative notion of entertainment of the player in predator/prey games. The methodology was based on qualitative considerations and empirical assumptions of what is enjoyable in such games and our focus was directed to the contribution of the behavior and strategy of game opponents in generating interesting games. A mathematical formula that determines an estimate of the real-time interest value of any predator/prey game was derived as the outcome of this method.

Herein we also discussed the generality of the interest metric proposed, which will be demonstrated experimentally in the following chapters. Moreover, some concerns and thoughts about the assumptions underlying the metric are raised here. Further analysis is presented in the discussion chapter of this dissertation (see Chapter 9). In the next chapter we provide background knowledge of the field (AI in games), the tools and methodology used in this dissertation.

Chapter 3

Background

3.1 Games

According to Wikipedia web-encyclopedia (2005), a game is a recreational activity involving one or more players, defined by a) a goal that the players try to reach, and b) some set of rules that determines what the players can do. Games are played primarily for entertainment or enjoyment, but may also serve an educational or simulative role.

Accordingly, a computer game is a concept inspired by the real world and composed of a computer-controlled virtual universe that players interact with in order to achieve a defined goal or set of goals. In that sense, board or card games, conceptually designed to be played in the real world, that are played by digital means are excluded from the investigations of this dissertation. Additional features of the test-bed computer games investigated are: 1) they are played off-line (no web access) 2) they are played in a personal computer; a single CPU is all the processing power available.

3.2 AI and Games

One of the primary goals of AI is to produce intelligent physical agents, i.e. robots. Brooks (1990) suggested that the path towards this goal starts from experiments on the low-level research of the real-world and continues with high-level decision making research on simulation. However, as noted in (Etzioni, 1993), these two directions can be followed in parallel. Computer games offer an ideal arena that combines AI research

in complex simulated worlds with an application of high commercial potential (Funge, 2004).

Over the last 25 years there have been major steps forward in computer games' graphics technology: from abstract 2D designs to complex realistic virtual worlds (Terzopoulos *et al.*, 1994) combined with advanced physics engines (Bourg, 2001); from crude character representations to advanced human-like characters. Meanwhile, sophisticated AI techniques (e.g. machine learning) in computer games are nowadays still in their very early stages, since computer games continue to use simple rule-based finite and fuzzy state machines for nearly all their AI needs (Cass, 2002). All these are well supported by game developers' statements that we still meet newly released games with the same 20-year old concept in brand new graphics engines (Woodcock, 2001).

In the mid-nineties, game developers began to argue for more AI in their games. The state of computer games at that time was stagnating – the games were very often derivative clones of one another as well as becoming ever more market-driven and expensive to develop. As Crawford would claim, "the graphics, animations and sound are better; the games have more internal detail, larger worlds, more complexity, but the basic designs have not changed over the last ten years" (Crawford, 1994; 1996).

From another viewpoint, the explosion of multi-player on-line gaming over the last few years it might indicate the increasing human need for more intelligent opponents. This also reveals that interactive opponents can generate interesting games, or else increase the perceived satisfaction of the player. Moreover, machine learning techniques are able to produce characters with intelligent capabilities useful to any game's context (Champandard, 2004). Therefore, conceptually, the absolute necessity of artificial intelligence techniques and particularly machine learning and on-line interaction in game development stems from the human need for playing against intelligent opponents. Game players seek continually for more enjoyable games as they spend 3.7 days per week playing an average of 2.01 hours per day (Rep, 2002), as stressed in (Fogel *et al.*, 2004), and this interest should somehow be maintained. Michael van Lent and John Laird (1999) state that an AI engine would be essential for reactive, context specific, flexible, realistic and easy to develop game characters. Nareyek (2002) also stresses the need for more intelligence in computer games.

It is only very recently that game industry has begun to realize the great (financial) importance of stronger AI in their products. Boon (2002) stresses that the most common complaint that gamers have is that the game is too short. However, as Boon claims, rather than making games longer game developers should focus on making games more interesting and appealing to both hard-core and soft-core gamers.

Unfortunately, instead of designing intelligent opponents to play against, game developers mainly concentrate on and invest in the graphical presentation of the game. We believe that players' demand for more interesting games and the increasing computational power will press towards an 'AI revolution' in computer games in the years to come. Some first signs of adaptive player modeling appear through the recently released *God genre* game 'Black & White' (Electronic-Arts, 2003) by Lionhead Studios (Molynoeux, 2001). In this game, the player controls a creature that learns, remembers and makes connections through Artificial Life techniques — the fundamental concept of 'Black & White' originates from the *Creatures* game developed in the late nineties (Cliff and Grand, 1999). Advanced machine learning techniques is the subsequent step to take. These techniques will be able to create the illusion of intelligence up to the level that is required by humans (Woodcock, 2001; Champandard, 2004; Funge, 2004).

3.3 Learning in Games

The majority of research on learning in games is built on board or card games. In the last decade many researchers have been involved in the development of intelligent opponents in those categories of games. Some of the attempts include evolutionary learning approaches applied from tic-tac-toe (Fogel, 1993) to checkers ((Fogel, 2002) among others), Go (Rosin and Belew, 1995; Richards *et al.*, 1998) and Monopoly (Frayn, 2005). In (Tesauro, 2002), a Temporal Difference Learning mechanism generates computer opponents capable of beating even expert humans in backgammon. These games are board games simulated in computers and therefore sometimes people refer to them as 'computer games'. However, when we refer to computer games we refer to the category of commercial games played by humans and non-player characters in virtual worlds.

Based on the success of the above-mentioned research on board games, the increasing

computing power and the commercial possibilities of computer games, very recently, researchers have attempted to introduce AI into computer games and have discussed the theoretical perspective of learning in different categories of games. Laird (2002) surveys the state of research in using AI techniques in interactive computer games. He also provides a taxonomy of games and the importance of computer games as experimental environments for strong AI application. Furthermore, Isla and Blumberg (2002) suggest potential research directions in AI game development, emphasizing to the emotional state and the perceived information of the character. Taylor (2000) attempts to bridge the gap between game development and modern AI by proposing artificial life techniques for generating physically modelled characters.

Game AI researchers, in their majority, focus on the genre of first-person shooter (FPS) games, primarily because of their popularity and secondarily because of their open source game engines. The most common FPS test-bed games are *Counter-Strike*, *Quake* and *Unreal Tournament*. Alex J. Champandard (2004) uses an FPS game to propose and apply a plethora of forms of AI techniques (varying from simple scripting to adaptive learning) for specific tasks like movement, shooting and weapon selection. His objective is to develop an open source AI game (so called FEAR project). Khoo (2002) developed an inexpensive AI technique based on the well known Eliza program (Weizenbaum, 1966) so that users get the impression of playing against humans instead of bots. In (Cole *et al.*, 2004), the parameters of the *Counter-Strike* built-in weapon selection rules are tuned by using artificial evolution. Furthermore, there have been attempts to mimic human behavior off-line, from samples of human playing, in a specific virtual environment. In (Bauckhage *et al.*, 2003) and (Thurau *et al.*, 2004), human-like opponent behaviors emerge through supervised learning techniques in *Quake*.

Other examples of learning in games primarily include reinforcement learning approaches in fight games (Graepel *et al.*, 2004); on-line learning for mobile phone games (Bjorsson *et al.*, 2004); learning through on-line interaction with synthetic characters (e.g. dogs) (Blumberg *et al.*, 2002; Dinerstein *et al.*, 2004); learning applied to low-level control (Manslow, 2002b). Alternatively, dynamic scripting and evolutionary learning has been used in a real-time strategy (RTS) game (Ponsen and Spronck, 2004).

There is a long debate on which form of learning is the most appropriate and feasible for a computer game application. On-line learning can be slow and lead to undesired and unrealistic behavior but it can demonstrate adaptive behaviors. Off-line learn-

ing is more reliable but it generally generates predictable behaviors (Manslow, 2002a; Champandard, 2004). However, researchers have shown that on-line learning in computer games is feasible through careful design and effective learning methodologies (Demasi and de O. Cruz, 2002; Johnson, 2004; Ponsen and Spronck, 2004; Stanley *et al.*, 2005).

Learning in non-game virtual worlds is a broad and active area of research that may very well interact with the research endeavors on AI in games. Researchers in that field are primarily focused on the behavior of simulated animals or artificial creatures. In (Grzeszczuk *et al.*, 1998) and (Terzopoulos *et al.*, 1996), studies on learning locomotion for dolphins and fishes are respectively presented. Sims (1994), on the same basis, uses evolutionary computation to configure the shape of virtual creatures that are able to walk.

3.3.1 Evolutionary Learning in Computer Games

Apart from the aforementioned evolutionary approaches in board or card games, evolutionary computation is not very well studied and explored in the area of computer games. In particular, the primary reason against its use for learning while playing (on-line) is its slow convergence and the fact that undesired/unpredictable behaviors may emerge. However, real-time adaptation which exhibits intelligent opponents is the main feature that motivates research on on-line evolutionary learning. Very recently, successful on-line neuro-evolution applications (Stanley *et al.*, 2005; Yannakakis and Hallam, 2005b) demonstrate the feasibility of the method through more efficient learning procedures and careful representation design.

Some few examples of evolutionary learning appear in (Champandard, 2004) where a genetic algorithm is used off-line to yield successful dodging-fire and rocket-jumping behaviors of NPCs of a FPS game. In addition, Ponsen and Spronck (2004) have used genetic algorithms off-line to design tactics for a RTS game. Moreover, co-evolutionary off-line and on-line approaches have been analyzed within an action game platform (Demasi and de O. Cruz, 2003). Blair and Sklar (1999) explore evolutionary learning techniques in a simulated single-agent Ice-Hockey environment. The Tron game is used by Funes and Pollack as a test-bed for applying co-evolution techniques against human players through the internet (1998; 2000). Their target is the emergence of human-like agent behaviors. Similarly, Fogel et al. have recently suggested a

game platform for testing on-line evolutionary methods for generating adaptive human-like realistic characters (2004). Other examples of evolutionary learning over various genres of game include the work of Togelius and Lucas (2005) for the emergence of foraging-related behaviors in the game named 'Cellz'.

Evolutionary learning of neural controlled NPCs is the core of the work of Yannakakis et al. (2004a; 2004; 2004b; 2005b). Among their contributions, a robust and highly adaptive on-line evolutionary learning mechanism is presented. The predator/prey genre of games is this approach's test-bed. In the same research direction, Stanley et al. (2005) are applying neuro-evolution techniques for the emergence of adaptive behaviors (e.g. capture-the-flag and wall-avoidance) in the 'NERO' training game in real-time. For this game, the NeuroEvolution Augmenting Topologies (NEAT) method (Stanley and Miikkulainen, 2002) has been used to evolve large populations of NPCs through an on-line replacement mechanism of the worst-fit NPCs (see also (Yannakakis *et al.*, 2004) for the introduction of the replacement methodology). In NERO, both the game genre (training) and the number of fifty NPCs contribute to the efficiency and the convergence time of the mechanism. Both leave space for unpredictable and/or unwanted emergent opponent behaviors to be accepted and/or ignored by the player. On the other hand, in (Yannakakis and Hallam, 2005b), on-line evolutionary learning is successfully applied in a computer game of four opponents where no credit is given by the player for such unrealistic behaviors.

3.3.1.1 Learning in Predator/Prey Games

Predator/prey games is a very popular category of computer games and among its best representatives is the classical Pac-Man released by Namco (Japan) in 1980. Even though Pac-Man's basic concept — the player's (*PacMan*'s) goal is to eat all the pellets appearing in a maze-shaped stage while avoiding being killed by four opponent characters named '*Ghosts*'— and graphics are very simple, the game still keeps players interested after so many years, and its basic ideas are still found in many newly released games.

Kaiser et al. (1998) attempted to analyze emotional episodes, facial expressions and feelings — according to the Facial Coding Action System (Eckman, 1979) — of humans playing a predator/prey computer game similar to Pac-Man (Kaiser and Wehrle, 1996). Other examples in the Pac-Man domain literature include researchers attempt-

ing to teach a controller to drive *PacMan* in order to acquire as many pellets as possible and to avoid being eaten by *Ghosts*. Koza (1992) considers the problem of controlling an agent in a dynamic non-deterministic environment and, therefore, sees Pac-Man as an interesting multi-agent environment for applying off-line learning techniques based on genetic programming. Other approaches, such as incremental learning (Gallagher and Ryan, 2003), and neuro-evolution (Lucas, 2005) have also been applied for producing effective Pac-Man playing strategies. The same Pac-Man application domain has been used for analyzing size and generality issues in genetic programming (Rosca, 1996).

On the other hand, there are many researchers who use predator/prey domains in order to obtain efficient emergent teamwork of either homogeneous or heterogeneous groups of predators. Luke and Spector (1996) have designed an environment similar to the Pac-Man game (the Serengeti world) in order to examine different breeding strategies and coordination mechanisms for the predators. Finally, there are examples of work in which both the predators' and the prey's strategies are co-evolved in continuous or grid-based environments (Miller and Cliff, 1994; Haynes and Sen, 1995).

3.3.2 Cooperation in Predator/Prey Worlds

Emerging cooperation in prey and predator artificial worlds is a field of reference for this work. One popular example of such work is the evolving neural network procedure used in Werner's *BioLand* (1993) simulated world to generate both herding and effective prey behaviors. Miller's and Cliff's work (1994) in co-evolution techniques of pursuit-evasion tactics and Koza's (1992) genetic programming work in a wide array of pursuit-evasion simulated scenarios also constitute characteristic pieces of work in the field.

3.4 Entertainment Metrics

As previously noted in Chapter 2, even though complex opponent behaviors emerge through machine learning techniques, there is no further analysis of whether these behaviors contribute to the satisfaction of the player. In other words, researchers hypothesize — by observing the vast number of multi-player on-line games played daily

on the web — that by generating human-like opponents (Freed *et al.*, 2000) they enable the player to gain more satisfaction from the game. According to Taatgen et al. (2003) believability of computer game opponents is strongly correlated with enjoyable games which are produced through cognitive models. These hypotheses might be true up to a point; however, since a notion of interest has not been explicitly defined, there is no evidence that a specific opponent behavior generates enjoyable games. This statement is the core of Iida's work on entertainment metrics for variants of chess games (Iida *et al.*, 2003).

Inspired by Iida's metric of entertainment, Yannakakis and Hallam (2004a) introduced a generic metric of entertainment for computer games (see also Chapter 2). This metric has been used for variants of predator/prey games and has been successfully cross-validated with human notions of entertainment (Yannakakis and Hallam, 2005d).

3.5 Tools

In this section we present the primary tools we used for the successful completion of the thesis' objectives presented in Chapter 1.

3.5.1 Evolutionary Computation

There are several approaches of evolutionary computation such as genetic algorithms (Holland, 1975), genetic programming (Koza, 1992), evolutionary strategies (Bäck and Schwefel, 1993) and evolutionary programming (Fogel *et al.*, 1966). All of them are population-based stochastic search algorithms that gain inspiration and principles from the Darwinian theory on natural evolution.

Evolutionary approaches are:

- Very good at dealing with large, complex search spaces which contain many local optima.
- Independent of gradient information.
- Able to deal with non-exact objective function problems.

All the aforementioned advantages make them very popular in multi-agent environment research and therefore, many researchers have tackled with evolutionary approaches in the control and management of agents.

3.5.1.1 Genetic Algorithms

An evolutionary approach called genetic algorithms (GAs) has been very popular among engineers and researchers since its first successful application (Goldberg, 1983). Holland (1975) was the first to show that a genetic algorithm can play the role of an adaptive system through artificial reproduction and evolution. Among their known advantages, GAs are very good in overcoming local optima towards the global optimum in multimodal complex search spaces.

The GAs' theoretical foundations as an optimization technique are based on the *schema theorem* (Holland, 1975). The schema theorem indicates exponential growth for consistently above-average-fitness schemas (a schema is a set of chromosomes that share specific values). Despite the vast variation of GAs used in the literature, there are some common basic algorithmic steps which are described as follows.

The GA maintains a population of chromosomes (or members of the population or solutions) which is randomly initialized. Then, at each generation, if a termination condition is not achieved (e.g. high fitness), do the following:

1. Evaluate each chromosome of the population by calculating its fitness as a solution to the problem under consideration.

2. Select parents based on their fitness.

3. Apply genetic search operators (mutation and crossover) to the selected parents and produce offspring which will form the next generation. Go back to step 1.

Genetic algorithms constitute the main genetic search method applied in this dissertation. Both generational and steady-state GAs (Goldberg, 1989) as well as advanced genetic operators like uniform crossover (Syswerda, 1989) and the Montana & Davis neural network (1989) crossover operator have been used in this work.

3.5.1.2 Estimation of Distribution Algorithms

Estimation of Distribution Algorithms (EDAs) is a new and prominent area of evolutionary computation (Larrañaga and Lozano, 2001). The algorithm used for the experiments presented in this thesis is based on the Univariate Marginal Distribution (Muehlenbein and Paass, 1996) for Continuous Domains (UMDA_c) (González *et al.*, 2002). This algorithm is used as an alternative evolutionary learning mechanism to the genetic algorithm approach.

Among the few existing literature UMDA_c applications we can distinguish the simple linear, and quadratic function approximations that appear in (González *et al.*, 2002). In addition, Bengoetxea et al. (2001) present a comparison between a UMDA_c and a steady state GA for image recognition. In this comparative case study UMDA_c appears much more efficient and faster than the GA approach.

The UMDA_c algorithm with tournament selection (N_T is the tournament size) works as follows. At each generation t, a population N_U of $n-$dimensional random variables $\mathbf{W}^t = (w_1^t, \ldots, w_n^t)$ is maintained. It is assumed that the joint probability distribution of $\mathbf{W}^t$ follows an $n-$dimensional normal distribution which is factorized as a product of n independent and unidimensional normal densities. Thus, each component of $\mathbf{W}^t$ is unidimensional, normal distributed, that is $w_i^t \rightsquigarrow N(\mu_i^t, \sigma_i^t)$, where

$$f_{N(\mu_i^t,\sigma_i^t)}(w_i^t) = \frac{1}{\sqrt{2\pi}\sigma_i^t} e^{-(w_i^t-\mu_i^t)^2/2(\sigma_i^t)^2} \tag{3.1}$$

with $i = 1, \ldots, n$ is the probability density function of a normal distribution with mean μ_i^t and standard deviation σ_i^t in point w_i. Table 3.1 presents the pseudocode for this algorithm. UMDA_c is used for evolving neuro-controllers (see Section 4.1.1.2) where the $n-$dimensional random variable $\mathbf{W}^t$ represents the connection weights of the artificial neural network (see Section 3.5.2).

3.5.2 Artificial Neural Networks

Artificial neural networks (ANNs) constitute universal approximation methods and therefore, given a big enough number of processing elements (neurons), they are capable of approximating any function (with a finite number of discontinuities) with arbitrary accuracy (Kurkova, 1991). This property is derived from, the much quoted

Table 3.1: Pseudocode for UMDA$_c$ with tournament selection

while no convergence **do**

begin

for $(j = 1; j \leq \mathrm{N_U}; j++)$

begin

Draw $\mathbf{W}^t$ to obtain N_T individuals:

$$\mathbf{w}^t_{1,j} = (w^{1,t}_{1,j}, \ldots, w^{n,t}_{1,j})$$

$$\mathbf{w}^t_{2,j} = (w^{1,t}_{2,j}, \ldots, w^{n,t}_{2,j})$$

$$\vdots$$

$$\mathbf{w}^t_{N_T,j} = (w^{1,t}_{N_T,j}, \ldots, w^{n,t}_{N_T,j})$$

Evaluate $\mathbf{w}^t_{1,j}, \mathbf{w}^t_{2,j}, \ldots, \mathbf{w}^t_{N_T,j}$

Select the best one:

$$\mathbf{w}^t_{(1:2:\ldots:N_T),j} = max\left\{ f(\mathbf{w}^t_{1,j}), f(\mathbf{w}^t_{2,j}), \ldots, f(\mathbf{w}^t_{N_T,j}) \right\}$$

end

for $(i = 1; i \leq n; i++)$

begin

Estimate the parameters of the new probability density functions

$$\mu_i^{t+1} = \frac{\sum_{j=1}^{\mathrm{N_U}} w^{i,t}_{(1:2:\ldots:N_T),j}}{\mathrm{N_U}}$$

$$\sigma_i^{t+1} = \sqrt{\frac{\sum_{j=1}^{\mathrm{N_U}} (w^{i,t}_{(1:2:\ldots:N_T),j} - \mu_i^{t+1})^2}{\mathrm{N_U}}}$$

end

end

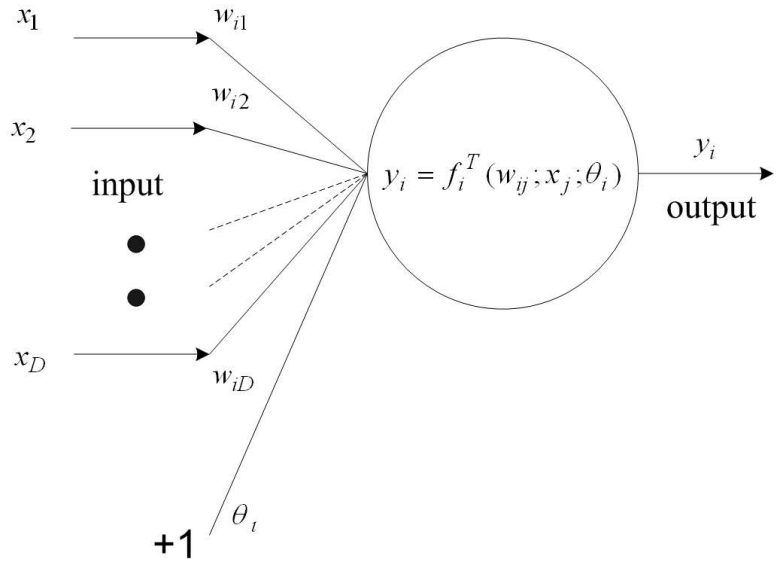


Figure 3.1: The artificial neuron.

and discussed in the neural network literature, Kolmogorov's (1957) superposition theorem.

The reader is advised to refer to (Hertz *et al.*, 1991; Haykin, 1998; Rumelhart *et al.*, 1986) for fundamentals on ANNs as well as state-of-the-art results in the field of *connectionism*, in which ANNs belong. In this section, we will go through a brief introduction on ANNs.

An ANN consists of a set of connected functional elements — also called neurons (see Figure 3.1) — and it is structured in a graph in which each node (i.e. neuron) i employs a transfer function f_i^T of the form

$$y_i = f_i^T\left(\sum_j w_{ij}x_j + \theta_i\right) \tag{3.2}$$

where y_i is the i^{th} neuron's output; x_j is the j^{th} input to the neuron; w_{ij} is the connection weight between neuron i and neuron j; and θ_i is the threshold (bias) of the neuron.

According to their connectivity, neural networks are divided into feedforward and recurrent. Regarding the recurrent connectivity we mention the Locally Recurrent Globally Feedforward (LRGF) networks as described in (Tsoi and Back, 1994) and the Elman networks (Elman, 1990) as being the most popular. Learning in neural networks is achieved by adjusting the connections' weights so that trained neural networks can

perform certain tasks. There are roughly three different types of learning. These are:

- **Supervised:** A subset of all possible input-output pairs is provided to the neural network, that is a 'training set'. A training cycle consists of the following steps. An input vector is presented at the inputs together with a set of desired responses, one for each neuron, at the output layer. As soon as the neural processes are complete, the errors or discrepancies between the desired and actual response for each node in the output layer are found. These are then used to determine weight changes in the net according to the prevailing learning rule.

- **Reinforcement:** Reinforcement learning (RL) is a type of supervised learning in the sense that some feedback from the environment is given. This feedback signal is only evaluative and not instructive and it is used to guide the neural network towards connection weight values that maximize rewards obtained. Reinforcement learning is often called 'learning with a critic' or 'learning with a teacher'. Among the various reinforcement learning approaches the most popular is Q-learning (Sutton and Barto, 1998).

- **Unsupervised:** No training set is available and no reinforcement is provided. In this type of learning the ANN is provided with an input pattern and it self-organizes in order to find the natural structure inherent in the input data. There are a number of unsupervised learning schemes, including competitive learning, adaptive resonance theory and *self-organising maps* (SOM). A popular type of SOMs is the Kohonen network (Kohonen, 1997).

There are various types of learning rules that may be applied to adjust the ANN's connection weights. The most commonly used are:

- **Hebbian rule:** This rule is based on Hebb's (1949) observation from neurobiological experiments: if neurons on both sides of a synapse (i.e. connection in a ANN) are activated synchronously, then the strength (i.e. connection weight in a ANN) of this synapse is increased.

- **Delta rule:** This learning rule adjusts the connection weight vector in the most efficient way as far as single-layer feedforward ANN are concerned. It performs a gradient descent towards the minimization of the error between the desired and the ANN output.

- **Backpropagation (BP):** A learning algorithm for multi-layer neural networks

was first introduced by Rumelhart, Hinton and Williams and named as 'back-propagation training algorithm' — also referred to as the generalized delta rule (Rumelhart *et al.*, 1988). At the output layer, the output vector is compared to the expected output. Subsequently, the error is calculated from the delta rule and is propagated back through the network. The idea which is based on the delta rule, is to adjust the weights to minimize the difference between the real output and the expected output. Such networks can learn arbitrary associations by using differentiable activation functions.

Since our aim is to emerge complex and adaptive behaviors within multi-agent environments, neural networks are considered to be a very efficient tool to control the agent's behaviors (Ackley and Littman, 1992).

3.5.3 Evolving Artificial Neural Networks

Learning by means of evolutionary computation (i.e. evolutionary learning) has been widely used type for automatically generating efficient ANN. Designing adaptive ANN controllers for agents in an unpredictable and continuously changing multi-agent environment such as a computer game can be a challenging task. As stressed in (Yao, 1995; Yao and Liu, 1996; Yao, 1999) Evolving Artificial Neural Networks (EANNs) is a learning mechanism that can successfully adapt to such an environment as well as to changes in it. Various case studies of the aforementioned work have show that one of EANNs' distinct features is their adaptability to dynamic environments. Thus, if in a sense, EANNs can be regarded as a general framework for adaptive systems, they consist of a strongly recommended tool for our computer games applications.

EANNs involve evolution of:

- Neural network architectures (Frean, 1990).
- Learning rules (Hinton and Nowlan, 1987).
- Connection weights (Whitley *et al.*, 1990; Belew *et al.*, 1992).

EANNs are used to evolve both the architectures and the connection weights of our game character ANN controllers. As far as the training of ANN weights is concerned, EANNs manage to overcome known drawbacks regarding the use of gradient descent. BP techniques are often trapped in local optima when dealing with large, complex and

multimodal search spaces. Moreover, supervised learning of ANN requires an appropriate set of training data which is sometimes either not available or very difficult to obtain (see Chapter 4 for such a case study). Because of the evolutionary approaches' effectiveness in such spaces, research on ANN training by means of artificial evolution (neuro-evolution) has been very active (see (Yao, 1999) for an extensive literature review).

3.6 Methodology

3.6.1 Emerging Cooperation

According to Funge (2004), computer games provide a perfect environment for research on emerging cooperation because they are based on simulation of highly complex and fully-dynamic multi-agent worlds. In contrast with the real world (i.e. realistic robotic environment), experiments can be easily observed and access to the world state is fully controllable.

Considerable research has been conducted towards the emergence of global cooperative behaviors from local communication in multi-agent environments. As far as the task of pursuit in worlds of multiple predators is concerned it determines a case study of a spatial coordination problem. Artificial life approaches include Reynolds' work on *Boids* (1987), which is based on the use of local partial communication, towards the generation of global successful flocking strategies. Spatial coordination (flocking) in artificial multi-agent worlds grounded in behavior-based (Balch and Arkin, 1998) and artificial potential field (Khatib, 1986) approaches are reported in (Flacher and Sigaud, 2004) and (Flacher and Sigaud, 2002) respectively. Inspired by Reynold's work, the Icosystem Game (2002), designed by Eric Bonabeau, is a simple agent-based simulation that demonstrates emergent collective behaviors which are based on local interactions. Agents in this game are assigned an aggressor and a defender agent and they move to keep their defender between them and their aggressor. In (Parker, 1993), the proper levels of balance between global and local knowledge of a multi-agent simulated world are investigated. Cooperative flocking (or else "keep formation") behaviors emerge through local rules.

With artificial evolution present, Reynolds (1993) used genetic programming tech-

niques for the emergence of herding behaviors of *Critters* against predators. He evolved a motion controller gathering information about its neighbors and predators in a homogeneous environment. On the other hand, there have been difficult tasks, such as schooling behaviors in simulated fish, that did not manage to emerge (Zaera *et al.*, 1996). Parunak and Brueckner (2000) introduce a pheromone based approach to yield collective behavior in a decentralized fashion.

Similar work can be found in the *Serengeti* world of Luke and Spector (1996). They attempted to examine the correlation between breeding strategies, coordination mechanisms and performance of teamwork hunting behaviors. One of their conclusions is that homogeneous agents perform well as a cooperative group when they perceive information relatively to them (e.g. relative coordinates).

Previous work on evolutionary approaches in neural controlled agents includes the simulated world of Ackley and Littman (1992). Their evolutionary reinforcement learning model consists of adaptive artificial creatures that move randomly on a two-dimensional environment, encountering food, predators and other types of tasks. The set of their actions is controlled by two feedforward neural networks: (1) an evaluation network representing how good the agent's state is and (2) an action network that outputs the agent's action in each time step. Ackley and Littman showed that neural networks can easily exploit various form of learning and therefore, help and speed up the evolutionary process. Thus, in the question: "What to evolve?" the answer is neural networks because they appear to be the most promising way of emerging complex behaviors in environments such as a multi-agent game.

Seeing our work from the point of view of emerging complex behaviors in multi-agent simulated worlds, *Creatures* (Cliff and Grand, 1999) and *PolyWorld* (Yaeger, 1993) constitute related examples. Both simulated worlds involve multiple neural-controlled creatures that learn to achieve specific tasks on-line.

Artificial organisms and attempts towards the emergence of cooperative foraging behaviors amongst them is a field closely related to our work. A representative example of that field of research is AntFarm (Collins and Jefferson, 1992) which is a world used for investigation of cooperative neural network controlled artificial ants through evolution. Such approaches, however, are based on active (chemical) communication via pheromone trails which contrasts our game worlds' properties. For an extended overview of similar swarm intelligence techniques see (Bonabeau *et al.*, 1999) and

(Dorigo *et al.*, 1999).

3.6.2 Learning

The main features of the learning mechanisms used in the game environments can be demonstrated as two axes of research that this thesis attempts to cover. The basic axis (horizontal in Fig. 3.2) determines the type of learning and the mechanisms are distinguished between those that attempt to mimic a good behavior (or else 'learning by samples' — see Section 4.1.2) and those that reward good features of a behavior (or else 'learning by rewards' — see Section 4.1.1). Both types include evolutionary algorithms while learning by samples also include gradient search algorithms. The secondary axis (vertical in Fig. 3.2) determines the environment in which agents learn. This feature distinguishes mechanisms where agents learn individually and mechanisms where agents learn within a homogeneous group of their clones.

For this dissertation, when learning by rewards acts upon a heterogeneous environment, we call this 'learning by survival' since we explore heterogeneity through on-line processes. This is the type of learning (by the use of evolutionary computation) applied for the enhancement of the game's entertainment value (see Section 4.3). All aforementioned machine learning mechanisms are comprehensively presented in Chapter 4 and the homogeneous approaches are evaluated in a test-bed simulated world.

3.6.3 Human-Centered Experiments for Games

The human aspect of computer games has been very well investigated through various fields of research. A popular technique for the evaluation, testing and/or validation of any applied methodology interacting with humans is through surveys of statistically significant numbers of subjects (human players).

Livingstone and McGlinchey (2004) have introduced the so called 'believability tests' to measure whether an obtained AI opponent is believable by humans or not. Such surveys with human players attempt to bring the Turing machine (1950) principle to computer games by creating behaviors that appear intelligent (Rabin, 2002). On that basis, a human-centered approach to evaluate human characteristics in game characters is proposed by Norlig and Sonenberg (2002). In addition, a comparison between

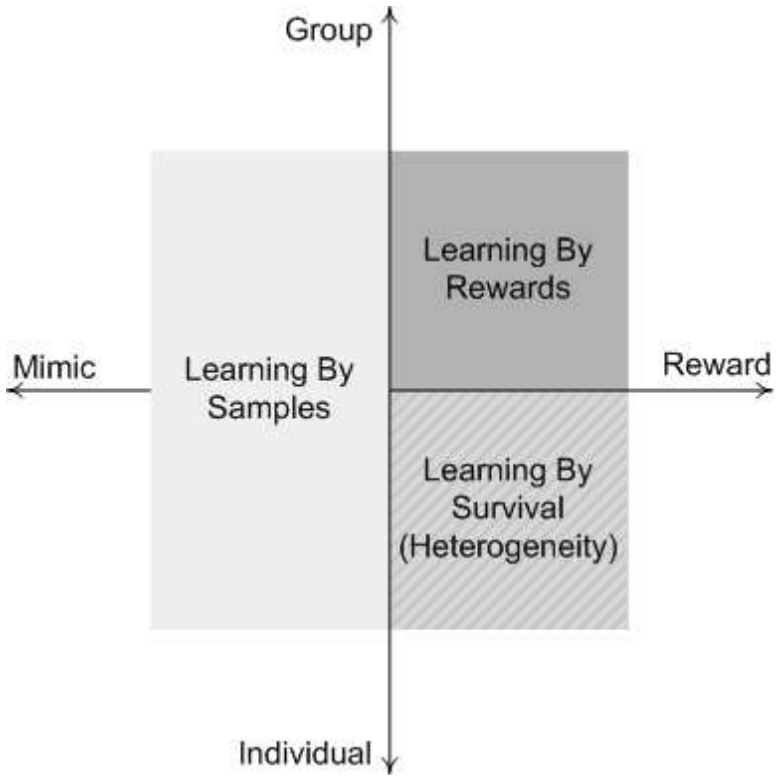


Figure 3.2: The two axes that determine our research focus and classify the learning mechanisms used.

heuristics regarding the 'playability' of computer games and human judgement of various game features is reported in (Desurvire *et al.*, 2004). Humans are used as samples for studying the emotional flow through real-time facial expressions and on-line questionnaires in (Kaiser *et al.*, 1998).

According to a study reported in (Sweetser *et al.*, 2003), a questionnaire administered to a group of university students was directed towards ascertaining the importance of different aspects of player behavior in computer games. It was found that people who prefer playing computer games with other humans tend to value intelligent behavior and social interaction more than people who prefer computer players. Accordingly, people who prefer computer players do so for convenience, practice and a preference for games that can only be played individually.

3.7 Complementary Review

Computer games is a highly interdisciplinary field of research. Apart from computer science and AI, areas such as psychology, sociology, education, graphics' design and arts are inspired and motivated by computer games. However, since we envisage a digital entertainment with richer and more 'meaningful' interaction, we will outline the research on the complementary fields of emotional psychology, sociology and education that constitute the humanitarian aspect of games.

There is a major direction in emotional psychology that focuses on the impact of computer games in learning. It addresses the element of entertainment as a powerful motivation tool (Hartmann and Rollett, 1994). See also (Burg and Cleland, 2001) for research on the benefits of computer games for a computer-enhanced education. As recent studies show, computer games could become part of a computer-enhanced education since there are indications that they can help students learn faster and more efficiently (Beal *et al.*, 2002; BBC, 2002b). Researchers in the U.K. looked into games like *Championship Manager* and *SimCity* and recognized their positive impact on teaching children how to think clearly and make decisions (BBC, 2002a). The social implications of computer games is also a hot topic of discussion ((Rabasca, 2000) among others) still dividing sociologists between those that are for and those against the use of computer games.

3.8 Summary

In this chapter the state-of-the-art of AI in computer games and the state of the computer games' industry was presented. Subsequently, the current research on machine learning applied in computer games was discussed and its potential was revealed. In addition, the literature review on the AI techniques; the basic steps of the methodology that this thesis covers; and some peripheral fields of interest were presented.

The following chapter illustrates comprehensively the learning methodologies used and introduces the first step towards applying the proposed techniques for obtaining off-line trained homogeneous teams of cooperative agents. Such teams of opponents will be used as starting points towards the on-line generation of games of higher entertainment in the subsequent chapters.

Chapter 4

Methodology

This chapter [1]presents the methodology followed for the successful completion of the thesis' aims. These include primarily the effective emergence of highly interesting computer games which partially derives from emergent cooperative behaviors built on minimal controller structures. On that basis, the learning procedures used to meet these objectives are described comprehensively and tested in a prototype two-dimensional, multi-agent, computer games-inspired simulated world.

4.1 Learning Cooperation in Multi-Opponent Games

As previously mentioned in Chapter 1, our first objective considering two-dimensional computer game worlds is to generate cooperative behaviors among the multiple opponents that appear. Cooperation is a feature that augments intelligence of the opponents and consequently improves the player's enjoyment. From that perspective, teamwork is a desired gaming opponent behavior.

The learning mechanisms used to generate cooperative behaviors are classified into supervised (or else 'learning by samples') and unsupervised (or else 'learning by rewards') according to whether there is a desired near-optimal spatial coordination behavior available to learn from or not (see also Section 3.6.2).

[1]Parts of this chapter have been published in (Yannakakis *et al.*, 2003; 2005a) and (Yannakakis *et al.*, 2005b)

4.1.1 Learning by Rewards

In this subsection we present the two different evolutionary computation off-line learning mechanisms used for the experiments in this thesis. Their common feature is the emergence of the desired behavior by rewarding homogeneous agents that achieve overall good performance on their given tasks. More comprehensively, a generational genetic algorithm (Holland, 1975) and a modified Univariate Marginal Distribution for Continuous Domains (UMDA$_c$) (González *et al.*, 2002) are the evolutionary learning variants of this type of learning.

4.1.1.1 Generational Genetic Algorithm (GGA)

A generational genetic algorithm is implemented, which uses an "endogenous" evaluation function that derives from the agents' actions in the environment and promotes and/or penalizes behaviors according to the agents' tasks. Agents that learn to behave in this fashion are fit enough to be considered as good solutions of the problem.

The neural networks that determine the behavior of the agents are themselves evolved. In the algorithm presented here, the evolving process is limited to the connection weights of the neural network. Evolving both connection weights and topologies simultaneously is a more advanced algorithm described in Section 4.2.

The evolutionary procedure used can be described as follows. Each agent has a genome that encodes the connection weights of its neural network. A population of N_p (we keep this number low because of the computational cost) neural networks is initialized randomly. Initial real values that lie within [-5, 5] for their connection weights are picked randomly from a uniform distribution. Then, at each generation:

Step 1 Every agent in the population is cloned N times (N being the number of agents in the game environment). These N clones are placed in the game and tested for an evaluation period e_p. The outcome of this test is to ascertain real-time data which will be used to assess the fitness of each agent (see Figure 4.1).

Step 2 Each agent i is evaluated via a group fitness function f_i. By this evaluation, we mainly promote N clones of the same solution capable of cooperating in order to successfully achieve a desired behavior. Due to this, efficient cooperative behaviors emerge within a homogeneous environment.

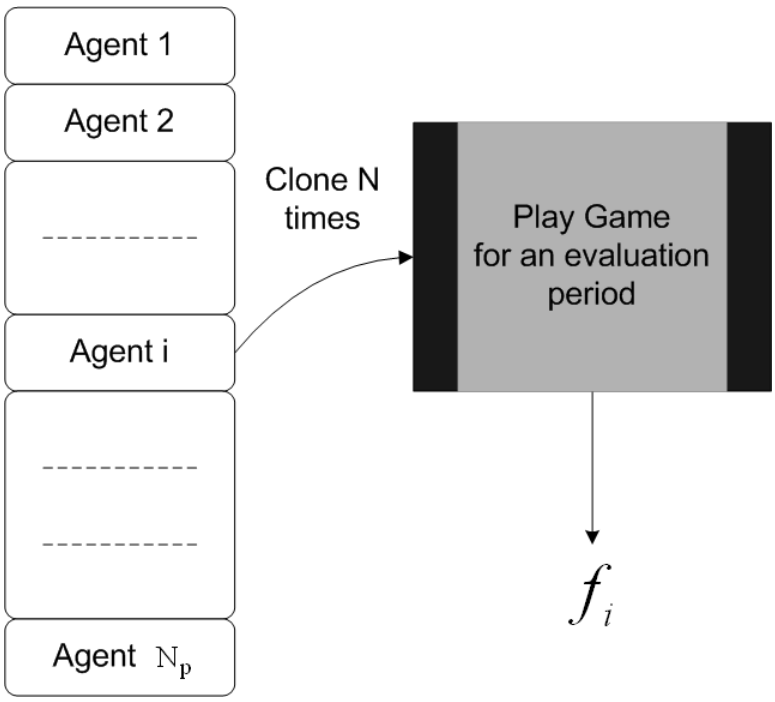


Figure 4.1: GGA: clonal evaluation of agents.

Step 3 A pure elitism selection method is used where the fittest N_s percent of the solutions is able to breed and, therefore, determine the members of the intermediate population.

Step 4 Each of the parents clones an equal number of offspring so that the total population reaches N_p members. Alternatively, uniform (Syswerda, 1989) and Montana and Davis (1989) crossover operators have been used at this step but proved unsuccessful. The explanation is the disruptive feature of crossover operators when dealing with distributed knowledge representation (i.e. neural network). That is, crossover among parts of different successful neural networks is very likely to lead into unsuccessful offspring (Yao, 1999).

Step 5 Mutation occurs in each gene (connection weight) of each offspring's genome with a small probability p_m. A uniform random distribution is used again to define the mutated value of the connection weight.

The algorithm is terminated when either a best fit agent is found or a large number of generations **T** is completed. We mainly use small simulation periods e_p to evaluate agents via f_i to limit the computational effort. Thus, this evaluation function constitutes an approximation of the overall performance of the examined agents in large simulation

periods. The higher the number of e_p simulation steps, the better the approximation of the agents' performance. Keeping an appropriate balance between computational effort and performance approximation is one of the key features of the GGA approach[1].

4.1.1.2 Univariate Marginal Distribution for Continuous Domains

Estimation of Distribution Algorithms have been previously introduced in Section 3.5.1.2. The EDA used for the game test-beds in this work is a modified Univariate Marginal Distribution for Continuous Domains (UMDA$_c$). This algorithm is used as an alternative evolutionary learning mechanism to the generational genetic algorithm approach presented in Section 4.1.1.1.

This algorithm, at each generation t, maintains an $n-$dimensional random variable $\mathbf{W}^t$, that represents the connection weights of the neural controller. We obtain a number of individuals N_T that defines the tournament size by drawing instances of the aforementioned $n-$dimensional random variable (i.e. connection weights). By using the GGA evaluation process (see Section 4.1.1.1), the fitness of these individuals is estimated and the best one is selected. By repeating this process $\mathrm{N_U}$ times we obtain a population of best fit selected individuals. This population is used to estimate the means and standard deviations of the random variable $\mathbf{W}^{t+1}$. These parameters are estimated by using their corresponding maximum likelihood estimators. Table 3.1 in Section 3.5.1.2 presents the pseudocode for this algorithm.

The UMDA$_c$ algorithm is terminated as soon as either a best fit set of connection weights $\mathbf{W}^t$ is found or a large number of generations $\mathbf{T}$ is completed.

4.1.2 Learning by Samples

In this subsection we present three supervised learning mechanisms used for our experiments. All of them attempt to yield desired agent behaviors by mimicking a fixed near optimal strategy. Thus, such mechanisms are only applied off-line and only when a near optimal spatial coordination strategy is available. Gradient-search back-propagation (BP); steady state GA (Syswerda, 1991) and real-time Teacher (i.e.

[1]The GGA approach can be seen as a $(\mathrm{N_pN_s} + \mathrm{N_p}(1-\mathrm{N_s}))$ Evolutionary Strategy (ES); however, the GGA name is kept since the crossover operator has been used in its initial version.

optimal strategy) GA are the supervised learning mechanisms that are explored in this thesis.

4.1.2.1 Back-Propagation

The use of this supervised learning approach is based on an evaluation which promotes any behavior that mimics a hand-coded near-optimal agents' strategy. The data set used for the supervised BP training of the neural controllers consists of inputs (perceptions) and actions of that near-optimal strategy and it is partitioned into training and validation portions for estimation of the generalization *mse* error. Early stopping methodology is used for avoiding overfitting.

For each BP case study, many different training and validation data sets have been used in order to determine a data set that produces the smallest generalization error and furthermore, the best performance achieved from the mechanism. We believe that a learning mechanism's efficiency and reliability are based on the overall effort made for achieving desired solutions. As far as the 'learning by samples' mechanisms are concerned, this effort includes the experimental selection of the most appropriate data set.

The Levenberg-Marquardt algorithm (Hagan and Menhaj, 1994) was used to train the neural controllers. This algorithm appears to be the fastest method for training moderate-sized feedforward neural networks and has given the highest performance training results among many other training algorithms employed.

The algorithm is terminated either when it converges to a good training mean square error (mse) value (e.g. this *mse* value depends on the topology of the network) or when the *mse* on the validation data set increases (i.e early stopping) or once a predefined large number of epochs is completed.

4.1.2.2 Steady state GA

The use of a steady state GA (Syswerda, 1991) supervised learning approach is based on an evaluation that promotes any behavior that mimics the near optimal strategy. Thus, exactly as in the BP approach, the data set (samples) used consists of inputs and actions of the desired near-optimal strategy. Steady state GA (SSGA) constitutes an

alternative supervised genetic search algorithm to the gradient search BP algorithm. The SSGA evolutionary procedure used can be described as follows.

A population of N_p agents is randomly initialized. Then, at each generation:

Step 1 Each agent is evaluated via a fitness function f_i' that corresponds to the mean square difference between the agent's and the near-optimal strategy's path (spatial coordination).

Step 2 Parents are selected for the next generation. A pure elitism method is used where only the fittest N_s percent of the population is able to breed. As in GGA, the choice of this selection method is made due to its distinct ability to accelerate the evolutionary procedure towards its convergence.

Step 3 Each parent either clones an offspring or mates with another randomly selected parent to reproduce two offspring by crossover. Crossover operators that have been used for the SSGA are: 1) Uniform Crossover (Syswerda, 1989) and 2) Montana and Davis (1989) crossover method as presented by Mitchell (1996).

Step 4 Mutation occurs in each gene (connection weight) of each offspring's genome with a small probability p_m. A uniform random distribution is used again to define the mutated value of the connection weight.

Step 5 Offspring are evaluated via f'.

Step 6 Offspring replace only less fit existing members of the population. Therefore, if an offspring is the least fit candidate member of the population, it is not included in the next generation.

The algorithm is terminated when either a good fit agent is found or a large number of generations **T** is completed.

4.1.2.3 Teacher-based GA

The Teacher-based GA (TGA) is a supervised learning evolutionary algorithm which attempts to generate well-behaved agents by rewarding the ones that follow the near-optimal strategy's (i.e. the Teacher's) good paradigm of behavior. Hence, every agent that is placed into the game environment is rewarded or penalized for its actions by its own Teacher. More comprehensively, at each simulation step: the embedded near-optimal controller (i.e. Teacher) generates an action, and the agent (i.e. trainee) con-

troller does too. The difference between them determines the evaluation of the trainee. The trainee controller's action is then applied.

The TGA learning approach is built upon the GGA approach (presented in Section 4.1.1.1). Both the GGA and the TGA follow the same algorithmic steps apart from Step 2 which is:

Step 2 Each agent is evaluated by the mean square difference between their clones' and their respective Teachers' paths.

In contrast with the BP and the SSGA algorithms, the TGA approach has the advantage of retrieving real-time simulation data from the whole group of agents instead of using a predefined data set. This way, the problem's designer spends minimal effort in obtaining appropriate training data sets. There is apparently a risk of retrieving insufficient or inappropriate real-time data; however, it can be decreased by repeating the learning attempt. Another crucial difference between TGA and the other supervised learning mechanisms is the learning environment. Agents in BP and SSGA are offline self-trained outside any simulated world whereas, in TGA agents are cloned and trained as a group in the game simulator.

4.2 Controller Minimization

Minimization of motion controllers offers several advantages. The smaller the controller, the better (easier) the understanding of its functioning by direct inspection. Additionally, the controller gets computationally more efficient and less expensive. Finally, the size and the structure of the minimized controller may provide an estimate of the task's complexity (Ganon *et al.*, 2003).

We have developed a new evolutionary algorithm, called ECWAS for designing artificial neural networks automatically, inspired by the EPNet system developed by Yao and Liu (1996). ECWAS is a modified constructive algorithm that starts with a minimal neural network (i.e. 1 hidden layer, 1 hidden neuron) and during the evolving process it adds new layers and neurons. Because pure constructive algorithms are susceptible to stick at structural local minima (Angeline *et al.*, 1994), the ECWAS algorithm allows deletion of layers and neurons as well.

The ECWAS (Evolving Connection Weights and Architectures Simultaneously) algo-

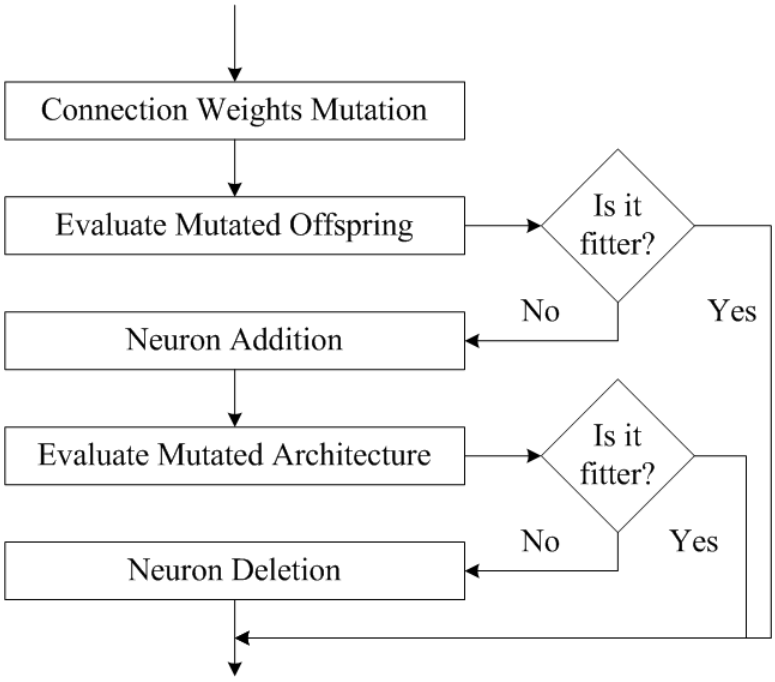


Figure 4.2: The structure of the ECWAS mutation operator.

rithm is built upon GGA presented in Section 4.1.1.1. The only difference between the two approaches is in the mutation operator used. As previously mentioned, the algorithm starts with a population of minimal neural networks having 1 hidden layer which contains 1 neuron. The modified mutation process (i.e. Step 5 of the GGA) contains three operators which occur in the sequence (see Figure 4.2):

Step 5a Connection weight mutation occurs as described in Section 4.1.1.1. The mutated offspring is evaluated via f and compared to its parent. If the offspring is fitter, then the mutation process is terminated, else go to Step 5b.

Step 5b A fully-connected hidden neuron (i.e. a neuron that is connected to all neurons or inputs of both its preceding and its following layer) is added to the network's current architecture. Both the neuron's connection weights and hidden layer are randomly selected from a uniform distribution. Once more, the mutated offspring is evaluated and compared to its parent. If the offspring is fitter, then the mutation process is terminated, else go to Step 5c.

Step 5c A randomly selected neuron as well as its connections are deleted. At this step there is no evaluation of the offspring as it is selected by default. This way, we try to bias the search toward minimal neural network architectures. The mutation

process is terminated at this step.

There is no upper bound for the number of hidden neurons in a hidden layer. On the other hand, the algorithm is constrained to design neural network architectures of the maximum number of three hidden layers. This constraint fulfils the well-known *Kolmogorov superposition theorem* (Kolmogorov, 1957) which states that there never needs to be more than three layers (i.e two hidden and the output layer) in a neural network to approximate any function.

ECWAS learning mechanism is used to automatically draw *a priori* near-optimal neural network architectures. As experiments showed, this algorithm constitutes an efficient pre-processing methodology for obtaining robust neural controllers of minimal size for the games used.

4.3 Interest Enhancement through On-Line Learning

We use an evolutionary machine learning mechanism for the games studied which is based on the idea of heterogeneous opponents that learn while they are playing against the player (i.e. on-line). The mechanism is initialized with some well-behaved opponents trained off-line and its purpose is to improve the entertainment perceived by the player. While 'learning by samples' and 'learning by rewards' mechanisms are devised to explore the emergence of cooperative opponents, this mechanism's purpose is to exploit the emergent cooperative features and further increase the player's entertainment through on-line interaction and adaptation.

The on-line learning mechanism is comprehensively described in the respective chapters of its game applications (see Chapter 5 and Chapter 6). However, its basic steps, which follow the GGA procedure, are presented briefly here as follows. At each generation of the algorithm:

Step 1: Each agent is evaluated every e_p simulation steps via an individual reward function, while the game is played.

Step 2: A pure elitism selection method is used where only a small percentage of the fittest solutions is able to breed. The fittest parents clone offspring.

Step 3: Mutation occurs in each gene (connection weight) of each offspring's genome with a small probability p_m.

Step 4: The mutated offspring is evaluated briefly in off-line mode, that is, by replacing the least-fit member of the population and playing a short off-line game of e_p simulation steps against a selected computer-programmed opponent. The fitness values of the mutated offspring and the least-fit member are compared and the better one is kept for the next generation.

The algorithm is terminated when a predetermined number of generations has elapsed.

4.4 Case Study

To study the emergence of cooperative behaviors within two-dimensional multi-agent game environments, we have developed a prototype simulated world called "*FlatLand*". *FlatLand*'s main properties collectively correspond to the game environment features defined in Chapter 1.

Our first objective in developing this world is to investigate the potential generation of cooperative complex behaviors amongst the agents given their type of communication and specific tasks they have to achieve. Subsequently, *FlatLand* is used to assess and compare the performance, robustness and effort cost of the off-line learning mechanisms described in this chapter. The two tasks that the agents are tested in are the antagonistic strategies of obstacle-avoidance and target-achievement. Overall, results show that cooperative behavior amongst the agents is necessary for the successful completion of their tasks. This behavior is built on implicit and partial communication. Moreover, the advantages of 'learning by rewards' methodology against 'learning by samples' in such game worlds are revealed.

The rest of this chapter is organized as follows. In Section 4.5, we present a detailed description of *FlatLand* as well as the agents' controllers employed. In Section 4.6, we discuss the difficulties and points of importance of this simulated world. The fitness functions used by the genetic-search learning algorithms are analytically described in Section 4.7. Results obtained in the 20-agent *FlatLand* world as well as comparison of performance, robustness and effort cost between the different learning approaches are presented in Section 4.8. Furthermore, experiments in more, and also less, complex environments are presented in the same section. The most important conclusions of the *FlatLand* research are summarized in Section 4.9.

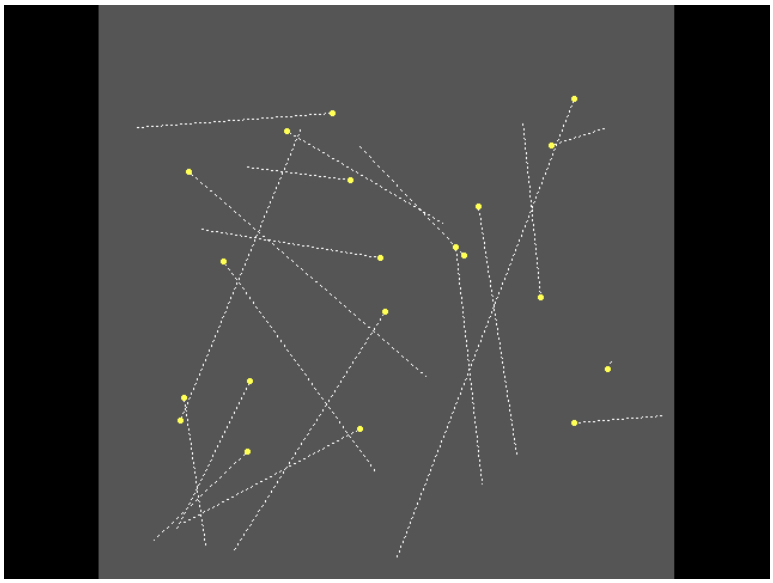


Figure 4.3: *FlatLand* world interface (the plane's dimensions are 80 cm $\times$ 80 cm for the experiments presented here).

4.5 The *FlatLand* Simulated World

The name "*FlatLand*" is inspired by the title of E. Abbott's book (1984) and its fundamental concept is based on previous research by Yannakakis (2001). Previous work on *FlatLand* is presented in (Yannakakis *et al.*, 2003). The main purpose of this simulated world is to be used as a test-bed environment for investigating evolutionary (Blair and Sklar, 1999) and gradient-based (to a lesser degree) learning techniques and furthermore, their ability to generate cooperative and complex obstacle-avoidance and target-achievement behaviors. In this section, we present a detailed description of this simulated world.

FlatLand is a square two-dimensional multi-agent environment. The world's dimensions are predefined (e.g. 80 cm $\times$ 80 cm) so that actions take place in a closed frictionless plane. There are two simple figures visualized in *FlatLand* (as illustrated in Figure 4.3): 1) white circles (radius of 5 mm) that represent the agents — artificial creatures; and 2) dashed straight lines connecting the agent's current position to its target point on the surface.

The population used consists of a number of 2D circular agent-creatures, called "Humans". The original case study of *FlatLand* institutes an environment of 20 agents (Yannakakis *et al.*, 2003) where each agent's motion is controlled by a neural network.

It is worth mentioning that one of Humans' properties is their permeability in case of a possible collision with each other. Therefore, their motion is not affected when they collide as they pass through each other. However, 'collisions' are penalized when assessing fitness.

Each Human is assigned a target point on the environment's surface. This point keeps changing during its life, hence as soon as a Human achieves its current target (i.e. manages to reach a circle of 5 mm around the target point), then a new target point is selected. The new target point is picked from a uniform random distribution at a specified distance of 30 cm from the agent's center. The simulation procedure of *FlatLand* can be described as follows. Humans are placed randomly in *FlatLand* via a uniform distribution. Then, the following occur at each simulation step:

1. Each Human gathers information from its environment (see Section 6.1.3.1).
2. It takes a movement decision (see Section 6.1.3.2).
3. Total number of collisions and target-achievements as well as the average speed and turn angle (see Section 6.1.3.2) of the Humans are recorded.
4. New randomly picked target points are given to those Humans that have achieved their target points.

FlatLand concentrates on the creation of emergent efficient and robust obstacle-avoidance and target-achievement behavior. Consequently, the design of the simulated agents used in this environment is deliberately kept abstract. Finally, there is no wall avoidance strategy implemented.

4.5.1 Neural Controller

Neural networks are a suitable host for emergent adaptive behaviors in complex multi-agent environments, as stressed by Ackley and Littman (1992); Yaeger (1993); and Cliff and Grand (1999). A feedforward neural controller is employed to manage the agents' motion and is described in this subsection. Apart from the neural controller, an Artificial Potential Field employed for controlling the agents' movement is also introduced in Section 4.5.2.

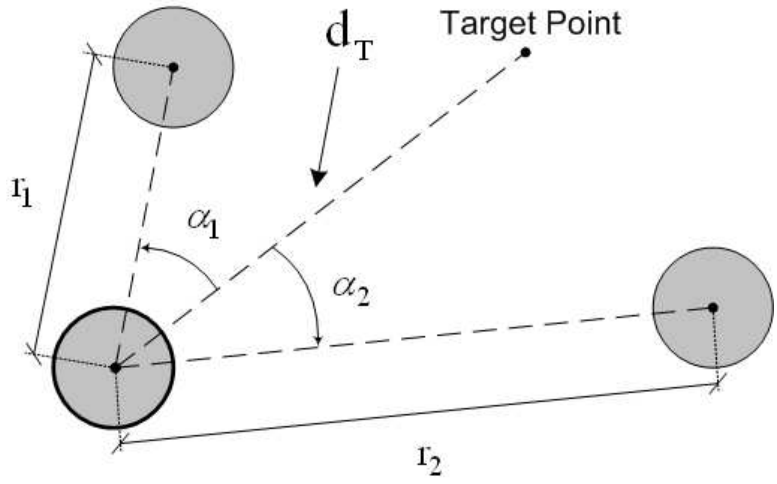


Figure 4.4: Human's input data in polar coordinates $(\mathbf{z} = 2)$.

4.5.1.1 Input

Using its sensors, each Human inspects the environment from its own point of view and decides about its next action. Sensors implemented are omni-directional with infinite range. Both the input information and the neural controller's architecture are analytically presented in this subsection.

The neural controller's input data and format can be described as follows. Each Human receives information from its environment expressed in the neural network's input array of dimension D:

$$D = 2\mathbf{z} + 1 \tag{4.1}$$

where $\mathbf{z}$ defines the number of the closest Humans that each Human perceives via its sensors. Thus, the input array consists of: (a) the polar coordinates (α_i, r_i) — based on the axis determined by the current position of the Human and its target point (see Figure 4.4) — of the $\mathbf{z}$ $(\mathbf{z} = 1, \ldots, (N-1))$ closest Humans sorted by distance (e.g. the polar coordinates of the closest's Human are inserted first in the input vector) and (b) an additional input that defines the distance between the Human's current position and its target point (d_T). Figure 4.4 illustrates the Human's sensor information as described above.

All input values are linearly normalized into [0, 1] before they are entered into the neural controller. The input format in polar coordinates is based on Reynolds' work on artificial critters (Reynolds, 1994). For the experiments presented in this case study $\mathbf{z} = 2$, which was found to be the minimal amount of information for a Human to

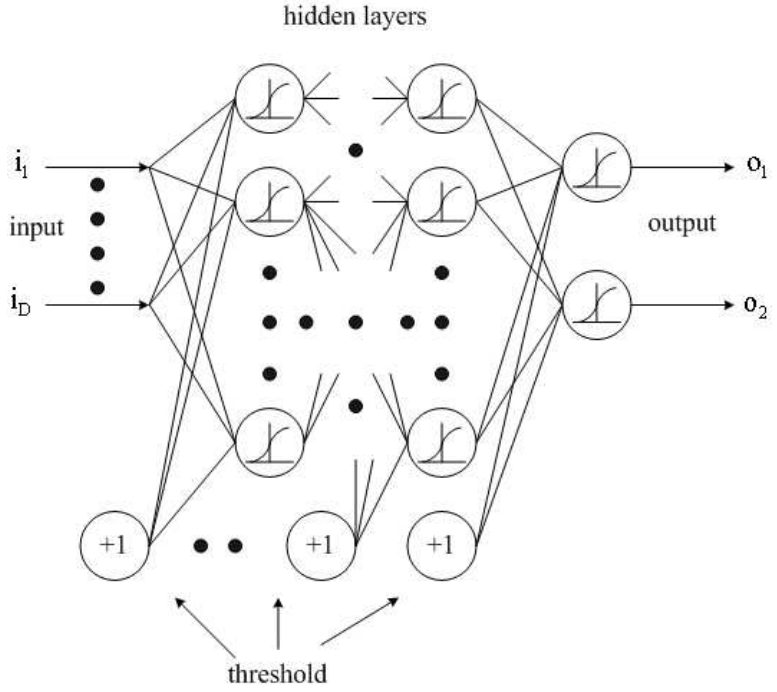


Figure 4.5: Multi-layer feedforward neural network controller.

successfully achieve the desired behavior (for $\mathbf{z} = 1$ neural controllers are not able to generate satisfactory obstacle-avoidance strategies).

4.5.1.2 Architecture

Our target when we first developed *FlatLand* was to find the simplest neural controller capable of generating the desired behavior. Since it is a quite challenging task to define and quantify simplicity of a neural network, we aim for the minimization of successful fully connected architectures (i.e. number of neurons and hidden layers). To this end, moderate size (i.e fewer than 2 hidden-layers and fewer than 30 hidden neurons in each layer) multi-layered fully connected feedforward neural networks (see Figure 4.5) have been used for the experiments presented here. The sigmoid function is employed at each neuron. In the attempt to minimize the controller's size, it is found (see Section 4.8.2.1) that single hidden-layered — containing fewer than 13 hidden neurons — neural network architectures are capable of generating efficient and robust solutions.

The connection weights take values from -5 to 5 while the neural network's output is a two-dimensional vector $[o_1, o_2]$ with respective values from 0 to 1. This vector

represents the Human's step motion and is converted into polar coordinates according to (4.2) and (4.3).

$$\mathrm{r_{NN}} = o_1 M \tag{4.2}$$

$$\alpha_{\mathrm{NN}} = (2o_2 - 1)\pi \tag{4.3}$$

where $\mathrm{r_{NN}}$ is the Human's step motion (in cm/simulation step); α_{NN} is the Human's turn angle from the axis determined by the Human's current position and its target point (in degrees); M is the Human's maximum speed — in experiments presented here, $M = 1$ cm/(simulation step).

4.5.2 Artificial Potential Field Strategy

Using the same environment, we explored an additional "species" of agents. These agents are called "Animals" and their only difference from Humans is in the control of their locomotion. Instead of a neural network, an Artificial Potential Field (APF), specially designed for this environment, controls the Animals' motion. The essence of the APF is that points along the Animal's path to its target point are considered to be attractive while obstacles (other agents) in the environment are repulsive (Khatib, 1986). The overall APF causes a net force to act on the Animal, which guides it along a collision-free, target-achievement path. For illustration, consider the Animal as a small sphere (of radius $R = 5$ mm) that slides down the surface plotted in Figure 4.6. This surface is plotted by each Animal at every simulation step and represents the function:

$$F(x,y) = \frac{M}{2}\sqrt{(x-x_{\mathrm{T}})^2 + (y-y_{\mathrm{T}})^2} \tag{4.4}$$

$$+\kappa \sum_{i=1}^{\mathbf{z}} e^{-\left[\left(\frac{\Delta x_i}{4R}\right)^2 + \left(\frac{\Delta y_i}{4R}\right)^2\right]} \tag{4.5}$$

where

$$\Delta x_i = x - x_i \tag{4.6}$$

$$\Delta y_i = y - y_i \tag{4.7}$$

$F(x,y)$ is the potential field value for the Animal's cartesian coordinates x, y; $[x_{\mathrm{T}}, y_{\mathrm{T}}]$ are the coordinates of Animal's target point; $[x_i, y_i]$ are the coordinates of the Animal's

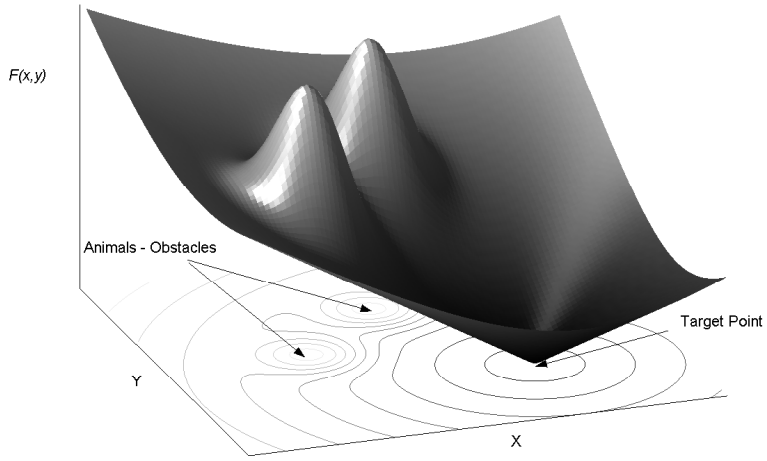


Figure 4.6: APF - Situation of two obstacles - closest Animals $(\mathbf{z} = 2)$.

i closest obstacle's (other Animal's) center; κ is a parameter that defines the height of the exponential "mountain-like" function presented in (4.5).

It is obvious that the surface plotted by each Animal alters at every simulation step as a result of *FlatLand*'s dynamics (moving obstacles — other Animals — and changing neighbors). The Animals' motion, thence, consists of a fixed non-linear strategy that does not evolve and is determined by the two-dimensional discontinuously time-varying potential field represented by (4.4). While, in theory, the APF solution may be prone to getting stuck in local minima, in practice, in the dynamic *FlatLand* world, the probability for such instances to occur is significantly low and, therefore, can be ignored.

Any motion strategy that guides an agent to quickly achieve its target, avoiding any possible collisions and keeping the straightest and fastest possible trajectory to its target, is definitely a "good" strategy in terms of *FlatLand* world. Hence, Animals present a "good" (near optimum) behavior in our simulated world and furthermore a reference case to compare to any Humans' behavior. This is the major reason for the use of this species of agent, along with the fact that data from the Animals' motion strategy can be used to train the Humans' neural network controller (see Section 4.1.2).

4.6 Hardness of the Problem

In this section we provide evidence of the problem's complexity and learning difficulty as well as its importance in the multi-agent systems area and this thesis in particular. In fact, *FlatLand* is a hard environment for an agent to learn to perform in because of its following distinct features:

- **Fully dynamical multi-agent**. Agents move continuously. Each agent faces a number of moving obstacles (i.e. potentially 19 other agents) in a specific squared environment and it has no *a priori* knowledge about their motion.
- **Partial information**. The fact that each agent in the *FlatLand* environment is able to capture the position of only $\mathbf{z}$ — in all experiments presented here $\mathbf{z} = 2$ — other agents adds the difficulty of partial information of the environment.
- **Implicit information**. An additional difficulty is that agents communicate just by "seeing" each other (see Figure 4.4). This kind of communication regarding the specific tasks (i.e. obstacle-avoidance and target-achievement) is very common in the animal world (e.g. predator-prey behaviors) as well as in human beings (e.g. crowded streets).
- **Discontinuous time-varying information**. The agent's input information suffers from discontinuity because of frequent alterations of the $\mathbf{z}$ closest neighbors that it takes into account via its sensors. Hence, the values of the polar coordinates α_i, r_i $(i = 1, \ldots, \mathbf{z})$ alter in a discontinuous fashion.
- **Supervised training**. If we try to train Humans under the near-optimum APF (Animals) strategy, we face problems of missing information for many instances. These are situations that Animals would never get into (e.g. the instance of $r_i \leq 2R$) but trained Humans do. Overall, the task of choosing the most appropriate training set is something of an art in itself that requires a lot of trial and error experiments. Unfortunately, the exact features for an efficient set of data cannot be explicitly defined. Such features include the tradeoff between size and computational effort, the right proportion of antagonistic behavior examples (i.e. target-achieving contra collision-avoidance examples) and the specific required examples for each trained behavior.
- **Very few collision examples**. One of the difficulties of the *FlatLand* world is the small number of collisions per simulation step in relation to the environ-

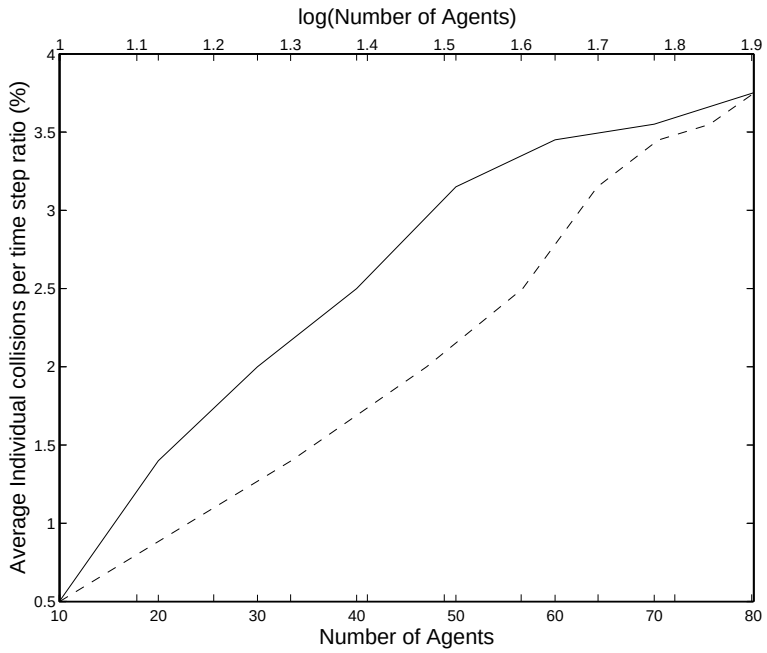


Figure 4.7: Worst collision-avoidance behaviors: average individual collisions per simulation step ratio over the number of simulated agents (solid line) and the logarithm of the number of simulated agents (dashed line).

ments' complexity. In the worst obstacle-avoidance behaviors we experienced, in the most complex environment of 80 agents, each agent collides 375 times in 10^4 simulation steps on average (i.e. 3.75% of its lifetime on average). For the simplest environment used (i.e. 10-agent) this percentage is approximately 0.5%. Therefore, it is both hard and computationally expensive for an obstacle-avoidance strategy to emerge from rewarding good examples of this strategy. Furthermore, when increasing the population of simulated agents, the average individual collisions per simulation step ratio appears to increase logarithmically (see dashed line in Figure 4.7).

FlatLand's basic concept and features make the proposed test-bed interesting for the multi-agent artificial life research area. The generality of this world extends into the area of computer games as successful applications have already shown (Yannakakis and Hallam, 2004a; Yannakakis *et al.*, 2004).

- **Emerging cooperation**. *FlatLand* is a simulated world in which we expect cooperative behaviors to emerge without any information exchange apart from spatial coordination (see above). Hence, emergent cooperation derives from 1) the way Humans move and 2) the way they interact with their environment (see

Section 4.8).

- **Strong creature-environment interaction**. There is a strong interaction and relation between the simulated creatures and their environment. In other words, any creature in *FlatLand* faces an environment of a two-dimensional space that includes a number of other creatures. Creatures in *FlatLand* are part of their own environment. Furthermore, *FlatLand*'s main feature, as an environment, is its own creatures. This feature defines an important point in the research of two-dimensional multi-agent dynamic simulated worlds. Computer games and artificial life offer a great arena of such worlds and a plethora of applications — see (Reynolds, 1987; Icosystem, 2002; Yannakakis and Hallam, 2004a; 2004b) among many.

4.7 Fitness Functions

As mentioned in Section 4.4, the *FlatLand* test-bed is utilized for investigation of evolutionary and gradient-based learning techniques and furthermore, their ability to emerge cooperative obstacle-avoidance and target-achievement behaviors. In this section we present the fitness measurements, that correspond to the *FlatLand* world and the above tasks, of all evolutionary learning mechanisms presented in Section 4.1.

GGA:

$$f_i = \frac{max\left\{1 - \frac{C_i}{C_u}, 0\right\} + min\left\{\frac{T_i}{T_u}, 1\right\}}{2} \tag{4.8}$$

where f_i is the evaluation function of Human i; C_i is the total number of collisions of Human i's N clones; C_u is the total number of collisions' upper bound which is determined by the total number of collisions of N "Target Achievers" (TAs) (i.e. agents that move directly towards their target points with constant speed — $\alpha_{NN} = 0^o, r_{NN} = 0.5$ cm/simulation step) in e_p simulation steps (for N=20 and $e_p = 300$, $C_u = 60$); T_i is the total number of target achievements of Human i's N clones; T_u is the total number of target achievements' upper bound which is determined by the total number of target achievements of N Animals in e_p simulation steps (for N=20 and $e_p = 300$, $T_u = 96$).

By using (4.8), we reward Humans (their N clones) that do not crash and achieve a determined number of targets (T_u) during an evaluation period. By this evalu-

ation, we mainly promote clones capable of cooperating in order to successfully achieve the aforementioned desired behavior.

SSGA:

$$f_i' = \frac{\sum_{t_s=1}^{S_t} \left\{ \left(o_{1,i}^{t_s} - \mathrm{r}_A^{t_s} \right)^2 + \left(o_{2,i}^{t_s} - \alpha_A^{t_s} \right)^2 \right\}}{S_t} \tag{4.9}$$

where f_i' is the evaluation function of Human i; $[o_{1,i}^{t_s}, o_{2,i}^{t_s}]$ is the neural network's output of Human i at step t_s; $[\mathrm{r}_A^{t_s}, \alpha_A^{t_s}]$ are the normalized polar coordinates (i.e. distance and angle into $[0,1]$) of the Animal's center at step t_s (Animal path); and S_t is the size of the training data set — for the experiments presented here $S_t = 666$.

This function represents the mean square difference between an Animal and a Human path (i.e. the same *mse* function that the BP attempts to minimize). In other words, it penalizes Humans that do not follow the near-optimal path (data set) of the Animals' (APF) strategy.

TGA:

$$f_i'' = \frac{\sum_{j=1}^{\mathrm{N}} f_{ji}'}{\mathrm{N}} \tag{4.10}$$

where f_i'' is the evaluation function of Human i; f_{ji}' is the mean square difference between the Teacher's (Animal's) and clone j's path (see (4.9)); and N is the number of clones placed into the *FlatLand* world — for the TGA experiments $\mathrm{N} = 80$.

4.8 Experiments

In this section we present and compare results obtained from all learning mechanisms applied in *FlatLand* as presented in Section 4.1. In particular, in Section 4.8.1 we present a way of evaluating the performance of any experiment, in Section 4.8.2 we introduce a methodology to optimize the neural controller architecture and based on this we compare the performance, robustness and effort cost of the mechanisms in the 20-agent *FlatLand* environment. We expand our experiments in decreasing and growing complexity environments (Section 4.8.3).

4.8.1 Performance Measurement

We introduce an efficient method for testing and comparing the learning mechanisms' ability to obtain successful controllers. For each learning mechanism used, we pick up the best (in terms of the optimization function used) neural controller (Human). Then, we record the total number of both collisions C and target achievements T of a population of N (e.g. N = 20) copies of this agent in a specific number of simulation steps (e.g. 10^4 simulation steps which take approximately 10 sec on a CPU of 1GHz) by placing these agents in *FlatLand* and running the simulation.

Since the initialization phase picks random numbers for initial positions and target points of the agents, it constitutes an important factor for any result. Therefore, we repeat the same procedure for ten simulation (i.e. evaluation) runs (we believe that this number of evaluation runs is adequate to illustrate a clear picture of the behavior) of different initial conditions and we compute the numbers of total collisions C_i and target achievements T_i for each run i. In addition, the agents' mean speed $E\{V\}$ and mean absolute turn angle $E\{a\}$ in degrees are calculated. Subsequently, the performance P_i of a team of agents in a single trial i is obtained as follows. We used 10^4 simulation steps for measuring and evaluating any behavior (collisions, target achievements) since we believe it is a sufficient period for evaluating a behavior of a population of agents in an efficient way. Subsequently, the performance P_i of a team of agents in a single trial i is given by (4.11)

$$P_i = \frac{max\left\{1 - \frac{C_i}{C_{TA}}, 0\right\} + min\left\{\frac{T_i}{T_A}, 1\right\}}{2} \tag{4.11}$$

where C_{TA} is the total number of collisions of N TAs in 10^4 simulation steps (for N = 20, $C_{TA} = 2000$ — see Table 4.3); T_A is the total number of target achievements of N Animals in 10^4 simulation steps (for N = 20, $T_A = 3200$ — see Table 4.3). The average performance over the ten trials is denoted by P.

The maximum value of (4.11) is 1.0 and it is obtained only when the agents do not collide at all and achieve as many target points as the Animals do (T_A) or more. Additionally, the upper bound for the total number of collisions is the number that the Target Achievers (TAs) produce (C_{TA}) because they just move directly towards their target points and therefore, present the worst collision-avoidance behavior from our viewpoint — even though randomly generated agents may produce more collisions

(see Table 4.3). Hence, (4.11) produces a clear picture of how far the performance of each learning mechanism is from the near-optimal performance of Animals ($P = 1.0$).

4.8.2 20-Agent FlatLand Environment

Experiments presented in this subsection are tested in the 20-Agent *FlatLand* environment (i.e. N = 20). This environment constitutes the fundamental test-bed for every investigation in *FlatLand*.

4.8.2.1 Optimal 1-Hidden Layer Neural Architecture

In order to efficiently compare every learning mechanism employed in *FlatLand* there first is a need for optimally designing the architecture of the neural controller. As previously mentioned in Section 4.4, one of our objectives in the *FlatLand* world research is the minimalization of the neural controller. Working towards this direction we use a modified version of the ECWAS algorithm which constrains the search to 1-hidden layer. Even though this modification decreases the search space, it does not significantly affect the overall performance of the produced behavior. This statement is based on experimental conclusions that even 1-hidden layer neural architectures can produce behaviors of high performance (see Section 4.8.2.2). By using this algorithm we attempt to find minimal neural topologies for solving the *FlatLand* problem as well as to avoid overfitting problems of the supervised learning mechanisms employed.

We experiment in the 20-agent *FlatLand* environment by applying the following procedure: a) repeat the modified ECWAS learning attempt (run) forty times (each time, a different random initialization of the connection weights' values is given); b) measure the performance of each run (see Section 4.8.1); c) for each neural architecture produced by the modified ECWAS algorithm calculate the number of runs that present higher performance than specific performance threshold values (i.e. $P > P_{th}$). This number determines the successes of the neural architecture for this performance threshold. The higher the performance threshold value, the more demanding the procedure.

Figure 4.8 illustrates the outcome of the aforementioned procedure. It presents the frequency of high performance (i.e. for 3 thresholds of performance: $P > 0.8$, $P > 0.85$ and $P > 0.9$) 1-hidden layer architectures found by the modified ECWAS algorithm. ECWAS results presented in Figure 4.8 show that the modified algorithm tends to

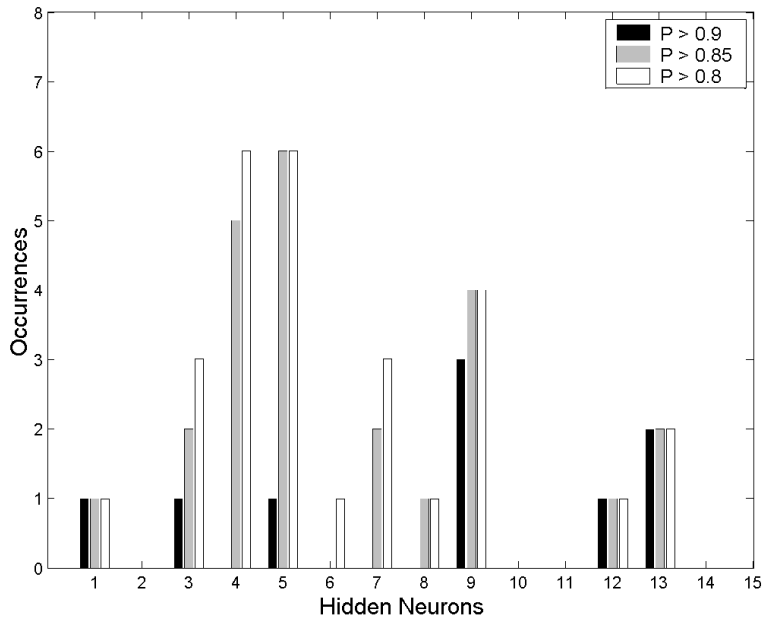


Figure 4.8: Occurrences of 1-hidden layer architectures found by the ECWAS algorithm.

find several different moderate size neural network architectures (i.e. fewer than 14 hidden neurons) capable of generating high-performance behaviors ($P > 0.9$). In order to choose the optimal among these successful neural controllers we sort them in frequency order and calculate their occurrences' mean performance and variance (presented in Table 4.1). We empirically select the architecture with 5 hidden neurons based on frequency, performance and robustness criteria. This architecture gets the highest mean performance and the lowest variance between its occurrences (i.e. most robust architecture). Even though, the neural architecture with 4 hidden neurons is the most frequent, it does not produce high-performance results ($P > 0.9$, see Figure 4.8), hence it is not selected.

To ensure that the selected architecture defines the optimal neural structure for gradient based algorithms as well, we introduce the following procedure. For every moderate sized (i.e. fewer than 15 hidden neurons) 1-hidden neural architecture a) repeat the BP run ten times; b) measure the performance of each run (see Section 4.8.1); c) calculate the mean performance and the variance over the ten runs. Results obtained from this procedure show that the most efficient and robust neural network architecture is again the one containing 5-hidden neurons. That is because this neural network architecture achieves the highest mean performance ($E\{P\} = 0.6403$) of any 1-hidden layer architecture examined (see Figure B.1 and Table B.1 in Appendix B). Smaller networks are

Occurrences	Hidden Neurons	$E\{P\}$	$\sigma^2\left(\cdot 10^{-4}\right)$
7	4	0.8520	22.7
6	5	0.8862	3.5
5	9	0.8782	29.9
4	7	0.8286	72.1
4	13	0.6418	916.2

Table 4.1: Mean performance $E\{P\}$ and variance σ^2 of the five most frequent 1-hidden layer neural architectures found by ECWAS. Only the occurrences whose performance is higher than 0.8 appear in Figure 4.8.

not able to fit the data and therefore, produce low performance behaviors. On the other hand, larger networks tend to overfit the data set and produce bad generalizations. Note that, the aforementioned procedure can be applied only for moderate sized 1-hidden layer neural networks. For neural networks of two or more hidden layers it is computationally expensive to investigate all possible architectures. Therefore, the ECWAS algorithm is the preferred method for selecting near-optimal neural architectures of that size because of its ability to automatically design successful neural controllers.

The fully connected neural network with 5 hidden neurons in 1 hidden layer is the architecture used for the experiments presented in *FlatLand*. This controller proves to be the most efficient 1-hidden neural network architecture produced from both genetic (ECWAS) and gradient search (BP) algorithms.

4.8.2.2 Best Performance Comparison

Table 4.3 illustrates the best (in terms of performance) obtained results from all different learning mechanisms applied in the 20-agent *FlatLand* environment. The neural controller employed is a 5-hidden neuron feedforward neural network. Apart from the evidence presented in Section 4.8.2.1, this controller exhibits the best behavior (in terms of performance), among all 1-hidden layer feedforward neural controllers, for all learning mechanisms applied.

In Table 4.3 we introduce the best obtained performance of a species of agents called "Random" ($P = 0.0010$). These agents are randomly initialized Humans and the variance of their performance over the 10 evaluation runs σ^2 equals $12.03 \cdot 10^{-7}$. The

Learning Mechanism	Parameters
BP	$S_t = 666$ (Animal path data)
SSGA	$S_t = 666$, $p_c = 0.4$ (uniform crossover), $\mathrm{N_p} = 2000$, $\mathrm{N_s} = 50\%$, $p_m = 0.01$, $\mathbf{T} = 2000$
TGA	$e_p = 8$, $\mathrm{N} = 80$, $p_m = 0.01$
GGA	$\mathrm{N_p} = 20$, $\mathrm{N} = 20$, $\mathrm{N_s} = 10\%$, $e_p = 300$, $p_m = 0.01$, $\mathbf{T} = 2000$
UMDA$_c$	$e_p = 300$, $\mathrm{N_U} = 20$, $N_T = 8$, $\mathbf{T} = 2000$

Table 4.2: Experiment parameters for the 20-agent environment.

Learning	Agents	Collisions	Target Achievements	Speed	Turn Angle	Performance
No	Random	198620	7	0.46	174^o	0.0010
	TAs	2000	3348	0.50	0^o	0.5000
	Animals	0	3200	0.5	3.2^o	1.0000
By Samples	BP	663	3121	0.51	7.8^o	0.8219
	SSGA	1159	3098	0.50	9.8^o	0.6943
	TGA	1513	2890	0.51	3.8^o	0.5733
By Rewards	GGA	261	3376	0.9	44.5^o	0.9347
	UMDA$_c$	335	3350	0.9	42.3^o	0.9162

Table 4.3: Best performance comparison table — average values are obtained from ten evaluation runs (10^4 simulation steps each) of a 20-agent environment.

Random agents along with the Target Achievers and the Animals are presented in Table 4.3 for comparison to any emergent Humans' behavior.

It is obvious that the GGA approach ($P = 0.9347$, $\sigma^2 = 12.5 \cdot 10^{-5}$) gets much closer to the desired behavior (i.e. Animals) than any other learning mechanism or any other "species" of agents. In general, both approaches based on learning by rewards manage to produce very high performance behaviors ($P > 0.9$). On the other hand, the best supervised learning performance, which is achieved by the BP approach, equals 0.8219. This large performance difference between the two ways of learning derives from the evidence that all 'learning by rewards' approaches — in contrast with the supervised learning approaches that attempt to mimic the Animal's behavior — generate Humans that manage to keep a big distance from each other in order to avoid collisions (see Figure 4.9). Furthermore, they move with an almost maximum speed (i.e. $E\{V\} = 0.9$) to achieve as many target points as possible.

On the other hand, by observing simulations of supervised mechanisms' emerged Humans we get some interesting conclusions as well. The emerged Humans in their attempt to mimic Animals behave as a TA-Animal hybrid. Low performance values, small turn angle (i.e. $E\{a\} < 10^{o}$) and speed that approximates 0.5 cm/(simulation step) illustrate the supervised trained behavior.

These results lead to the important conclusion that simple evolutionary learning mechanisms can produce much better behaviors than those produced by exhausting supervised learning approaches in *FlatLand*. Such successful solutions manage to exploit cooperative behaviors built on the partial and implicit spatial communication amongst Humans.

4.8.2.3 Robustness Comparison

We are interested in obtaining a successful and robust learning mechanism with minimum efforts in our experiments. We can obviously experiment with parameter value adjustment of each method and therefore, be able to find more effective neural controllers (Humans) for the desired behavior. However, if a successful controller is determined with the lowest computing cost, the applied methodology can be recommended.

To determine the effort that each learning mechanism has required to obtain a desirable robust neural controller, we assume that a single independent experiment is

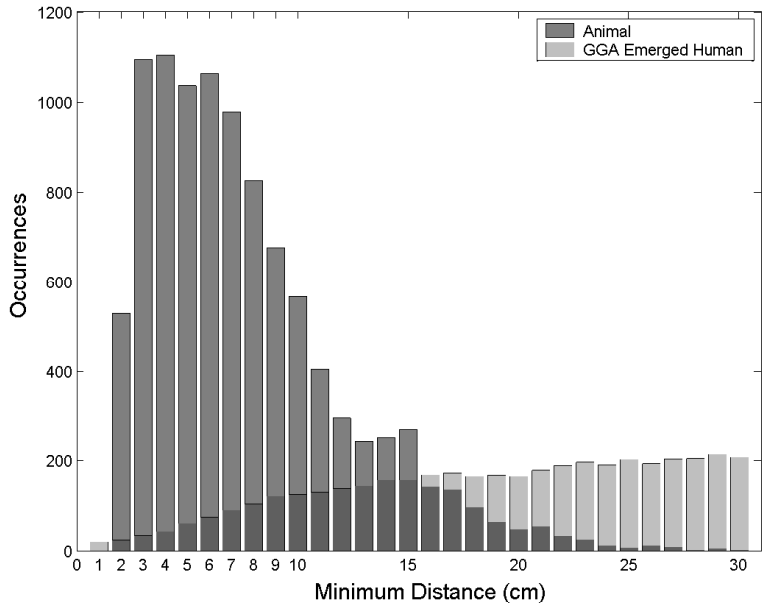


Figure 4.9: 20-Agent environment: Minimum distances' occurrences of an Animal and a Human emerged from GGA in 10^4 simulation steps. The dark gray color (not appearing in the legend), which indicates overlapping data, is produced due to the transparency of the colors used.

repeatedly run until a successful neural controller is found. A better mechanism will have a smaller number of runs to find a successful neural controller (Po Lee, 1998; Kim, 2002). To test the robustness of the solutions given and to calculate the effort cost of each approach, we apply the following procedure. For each approach a) repeat the learning attempt ten times; b) measure the performance of each run; c) calculate the successes of the approach for a specific performance threshold (i.e. number of runs that present higher performance than the threshold value). Figure 4.10 illustrates the number of successes of all learning mechanisms applied for ten values of P_{th}. The approaches' parameters are the same as the experiment parameters presented in Section 4.8.2.2.

The generational GA is the most efficient and robust approach for every performance function threshold (see Figure 4.10). Both GGA and UMDA$_c$ even generate controllers (3 success) with $P \geq 0.9$ whereas the supervised learning approach's best performance (BP) is below 0.85. 'Learning by rewards' approaches (GGA, UMDA$_c$) seem to be far more robust than any learning approach based on samples of good behavior (BP, SSGA, TGA).

UMDA$_c$ is a population based evolutionary algorithm that emerges as a generalization

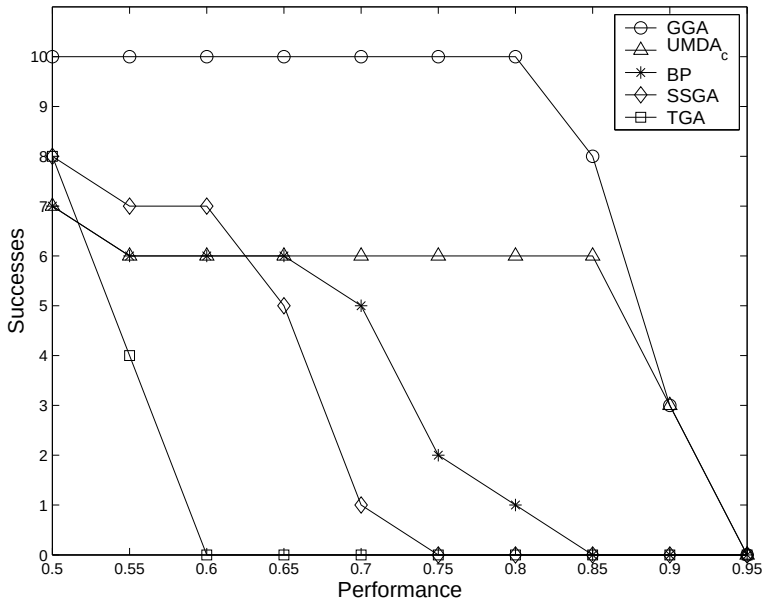


Figure 4.10: Number of successes out of 10 runs for specific performance values.

of the GGA approach for the purpose of overcoming poor performance in the specific problem. Instead of a genetic search by mutation, UMDA$_c$ approach searches through the solution's estimation of probability distribution in the search space. Despite its similarities to the GGA approach and its promise, it cannot compete with the robustness that the GGA demonstrates[2] (see Figure 4.10). Hence, it seems that the most appropriate evolutionary process for the *FlatLand* problem is based on pure genetic search.

It is worth mentioning that for $P_{th} = 0.7$, GGA is 100% successful (i.e. 10 out of 10 times), UMDA$_c$ succeeds 6 times while BP, SSGA and TGA succeed only 5, 1 and 0 times respectively. Thus, for $0.75 \leq P_{th} \leq 0.8$, the effort costs of GGA and UMDA$_c$ can only be compared with that of BP because SSGA and TGA fail completely (0 successes). Finally, for $P_{th} \geq 0.85$ there can be an effort cost comparison only between the GGA and UMDA$_c$ methods because they are the only two approaches capable of producing behaviors of that high performance.

[2] See also (Yannakakis *et al.*, 2005a) for an extensive comparative study between GGA and UMDA$_c$.
[3] in seconds

P_{th}	Approach	α	β	Confidence Interval	Effort Cost Interval[3]	Mean Effort Cost[3]
0.7	BP	5	5	[1.3051, 4.2772]	[122.14, 400.26]	205.87
	SSGA	1	9	[2.4225, 43.8596]	[7883.10, 142724.40]	35795.32
	GGA	10	0	[1.0023, 1.3984]	[797.96, 1113.30]	875.73
	UMDA$_c$	6	4	[1.2012, 3.2478]	[1353.99, 3660.93]	2066.53
0.75	BP	2	8	[1.9311, 16.6113]	[180.73, 1554.49]	514.69
	GGA	10	0	[1.0023, 1.3984]	[797.96, 1113.30]	875.73
	UMDA$_c$	6	4	[1.2012, 3.2478]	[1353.99, 3660.93]	2066.53
0.8	BP	1	9	[2.4225, 43.8596]	[226.70, 4104.39]	1029.38
	GGA	10	0	[1.0023, 1.3984]	[797.96, 1113.30]	875.73
	UMDA$_c$	6	4	[1.2012, 3.2478]	[1353.99, 3660.93]	2066.53
0.85	GGA	8	2	[1.0641, 2.0738]	[847.12, 1651.02]	1094.66
	UMDA$_c$	6	4	[1.2012, 3.2478]	[1353.99, 3660.93]	2066.53
0.9	GGA	3	7	[1.6402, 9.1491]	[1305.76, 7283.81]	2919.10
	UMDA$_c$	3	7	[1.6402, 9.1491]	[1848.78, 10312.90]	4133.06

Table 4.4: Effort cost comparison table ($\varepsilon = 0.05$) $Q_{\mathrm{BP}} = 93.58sec$, $Q_{\mathrm{SSGA}} = 3254.12sec$, $Q_{\mathrm{GGA}} = 796.12sec$, $Q_{\mathrm{UMDA}_c} = 1127.02sec$.

4.8.2.4 Effort Cost Comparison

Since 'learning by rewards' approaches (GGA, UMDA$_c$) are demonstrated to be more robust than any supervised learning approach used, the next step is to compare these mechanisms via their effort cost interval and mean effort cost. Hence, we pick decent high values of P_{th} (i.e. $P_{th} \geq 0.7$) and proceed with a beta-distribution approximation (see Appendix B) of the effort cost interval and the mean effort cost (Kim, 2002) for all approaches. Learning mechanisms that experience zero successes out of ten runs are not considered in further analysis.

Results from the effort cost comparison via the beta-distribution statistical method for five values of P_{th} are presented in Table 4.4. More comprehensively, for each P_{th} value the number of successes (α) and failures (β) of each approach is presented (as illustrated in Figure 4.10). By use of (A.2) the lower and upper bound probability χ_l, χ_u for each method is found; then the 95% confidence interval $[1/\chi_u, 1/\chi_l]$ is calculated. This interval represents the 95% confidence bounds on the expected number of runs required to achieve the first successful outcome. Table 4.4 also shows the effort cost interval $[Q_A/\chi_u, Q_A/\chi_l]$ for each approach, where Q_A corresponds to the unit computing cost per run of the approach A. For the experiments presented here Q_A equals to

the average CPU time of the ten runs (every experiment presented here ran in the same 1GHz processor). Finally, the mean effort cost is calculated with $\frac{\alpha+\beta+1}{\alpha}Q_A$.

The important conclusion that arises from Table 4.4 is that the BP approach is computationally preferred for low performance values (i.e. $P_{th} \leq 0.75$) from any other approach. In other words, if there is need for a fast, relevantly low performance solution (i.e. $P_{th} \leq 0.75$), the BP approach seems to be the most appropriate method. On the other hand, for $P_{th} = 0.8$ the learning mechanism's effort cost interval and mean effort cost show a computational preference for the GGA approach against the other two competing approaches (BP, UMDA$_c$). As previously stressed, only the GGA and UMDA$_c$ approaches are capable of producing high performance behaviors (see Figure 4.10). For such demanding solutions (i.e. $P_{th} \geq 0.85$), the GGA approach is proven to consume much less computational effort than the UMDA$_c$ approach does.

4.8.3 Growing FlatLand — Increasing Complexity

The effort cost analysis described in Section 4.8.2.4 presents a clear distinction of the BP and GGA approaches against the rest of learning mechanisms applied in the 20-agent *FlatLand* environment. In this environment the BP approach is preferred for efficiently generating relatively low performance solutions whereas the GGA approach is preferred for demanding high performance solutions.

However, in order to draw the overall picture of the aforementioned approaches' behavior in the *FlatLand* problem we need experimental results from less or more complex test-bed environments. Since the *FlatLand* problem becomes harder and more complex as the number of simulated agents increases, we pick the successful, in the 20-agent environment, BP and GGA approaches and test them in additional *FlatLand* environments of 10, 40 and 80 agents. Experiment parameters, performance and effort cost analysis for these environments are presented in Appendix B.

The conclusion that arises from Figure B.2 is the absolute supremacy of the GGA over the BP learning mechanism for every environment tested. GGA manages to get high performance behaviors ($P_{th} \geq 0.85$) even in the 80-agent environment whereas in the same environment BP generates behaviors of very poor performances (even though it is trained on animal data of this crowded environment). By observing the BP approach's behavior it gets obvious that the more complex the problem it gets, the harder the

neural network generalization becomes. There is definitely a significant difference among the two approaches' performance and successful runs in both the 10 and the 40-agent environment.

Given the obtained results and analysis, GGA seems to be the most robust learning mechanism applied in *FlatLand*. This mechanism's overall performance remains at very high levels even in very complex problems such as the 80-agent *FlatLand* environment (see Figure B.2(a)). Additionally, its required computational cost makes it preferable for high performance emergent solutions for every *FlatLand* environment (see Table B.3).

The major advantage of the GGA approach (against any supervised learning method) is that it always manages to produce a simple but highly effective emergent behavior. As previously seen in experiments from the 20-agent environment (Section 4.8.2), GGA generates Humans capable of staying far away from each other and moving at almost maximum speed. This cooperative behavior emerges in the other 3 environments examined in this section, as well. Cooperation emerges due to the fact that Humans are trained to behave as a group of homogeneous agents and is built on implicit and partial communication.

4.9 Conclusions

We introduced both a hard and interesting problem for the multi-agent dynamic simulated world research area. *FlatLand* shares common features of known artificial life and game worlds used for studying the emergence of cooperative global behaviors which are based on local interactions. In addition, agents are explicitly given individual tasks and their communication is limited to 'seeing' neighbor agents.

We saw that simple mutation-based evolutionary algorithms can generate robust and cooperative behaviors of high-performance as far as the complication of the *FlatLand* world is concerned. These algorithms' learning ability is based on rewarding the overall behavior of a group of clone agents (homogeneous team). More specifically, the GGA approach proved to be the most robust and less computationally expensive method for every *FlatLand* environment tested. On the other hand, supervised learning mechanisms failed to compete with the 'learning by rewards' approaches. Evidence supporting the obtained results are provided in the following section.

4.9.1 Why Reinforcement?

In this last section we discuss the appropriateness of machine learning mechanisms towards the emergence of cooperative behaviors in worlds that share properties with *FlatLand*.

Suppose that agents learn to behave within a group that consists of copies of themselves. This results in the emergence of an interesting form of abstract cooperation between the agent and its clones, which increases the global efficiency (performance). This is exactly what is demonstrated by all unsupervised learning approaches applied. On the contrary, in supervised learning an agent is initially self-trained (e.g. BP, SSGA) and then it clones itself to form a group. This procedure apparently does not leave any space for emergence of self-clone cooperation .

The question that arises here is: 'Is it the mechanism itself (unsupervised, supervised) or the learning environment (clonal, individual) that allows cooperation to appear and, therefore, produce such a big difference in performance, robustness and effort cost?' The answer is supported through the evidence that supervised learning even within homogeneous teams (i.e TGA approach) did not manage to compete with the 'learning by rewards' methods. Thus, it appears that reinforcing good solutions is the key factor that affects cooperation rather than the learning environment used. A point that further supports the aforementioned statement is that 'learning by rewards' approaches involve a minimal amount of data that the agents have to learn (e.g. the desired numbers of collisions and target achievements within an evaluation period).

On the other hand, no attempt to choose the most appropriate set of data (i.e. perception and action of Animals) for agents to learn from, proved able to outperform unsupervised learning. The complex dynamics of the environment and the strong training data set dependence constitute major obstacles towards a well performing solution. We saw that all mechanisms attempting to mimic the Animals' behavior managed to output sub-optimal (i.e. Target Achiever-Animal hybrid) behaviors. This is explained through the valid hypothesis that even slight differences from the near-optimal hand-coded strategy can cause collisions and therefore decrease the performance of a team of agents.

To summarize, suppose that a) we deal with fully dynamic multi-agent environments where the communication of agents is passive and based on partial and implicit information; b) we want the agents' controllers to learn to behave cooperatively for the

successful achievement of specific tasks; c) there is a near-optimal (good) hand-crafted behavior available; d) we have to make a choice between a supervised learning mechanism that attempts to mimic the good behavior and a mechanism that rewards the overall behavior of a group of agents and e) in both types of mechanisms the performance evaluation is based on the behavior of a homogeneous group of generated solutions (agents). Given these, a 'learning by rewards' approach tends to perform better by producing cooperative features within the emergent solutions. In addition, these solutions tend to be more robust and computationally preferable than solutions generated by mimicking.

4.10 Summary

Obtained conclusions from *FlatLand* constitute major input for the generation of collaborative opponents in games that share features with this abstract world. Given the shared features, the use of a moderate sized controller and the application of a GGA-based approach, it is shown that cooperation among opponents is plausible even when based on limited and implicit communication. The next chapter introduces the first experiments on a computer game by following the aforementioned guidelines in order to produce well-behaved cooperative computer opponents. These opponents are used as a starting point for the generation of enjoyable games.

Chapter 5

Dead End

In this chapter[1] we introduce the first stage of experiments on neuro-evolution mechanisms applied to predator/prey multi-character computer games. Our test-bed is a computer game where the prey (i.e. player) has to avoid its predators by escaping through an exit without getting killed. By viewing the game from the predators' (i.e. opponents') perspective, we attempt off-line to evolve neural-controlled opponents, whose communication is based on partial implicit information, capable of playing effectively against computer-guided fixed strategy players. However, emergent near-optimal behaviors make the game less interesting to play. We therefore introduce an entertainment measure for this specific game. Given this measure, we present an evolutionary mechanism for opponents that keep learning from a player while playing against it (i.e. on-line) and we demonstrate its efficiency and robustness in increasing and maintaining the game's interest. Computer game opponents following this on-line learning approach show high adaptability to changing player strategies which provides evidence for the approach's effectiveness against human players.

This chapter is organized as follows. In Section 5.1, we present a detailed description of the Dead End game (Park, 2003) as well as its characters' controllers. In Section 5.2, we discuss the difficulties of the problem as well as some issues of interest of our approach for the multi-agent computer games field. Then in Section 5.3, a methodology for adjusting interest measure parameters for the examined game is presented. Subsequently, Section 5.4 introduces a method for measuring performance in Dead End. The off-line and on-line machine learning mechanisms used are analytically described

[1] Parts of this chapter have been published in (Yannakakis *et al.*, 2004; Yannakakis and Hallam, 2005c) and (Yannakakis and Hallam, 2005a)

in Section 5.5 and Section 5.6 respectively. The game features of number of opponents and sensory information are discussed in Section 5.7. Off-line and on-line learning experiments are presented in Section 5.8 and Section 5.9. Finally, the most important conclusions of the Dead End research are summarized in Section 5.10.

5.1 The Dead End Game

In this section, we present a detailed description of the Dead End game and its two main characters, *Dogs* and the *Cat*. As previously mentioned, this game is investigated from the viewpoint of *Dogs* and more specifically how *Dogs'* emergent adaptive behaviors can be effective against skilled players as well as contribute to the interest of the game.

The Dead End game field (i.e. stage) is a two-dimensional square world that contains a white rectangular area named "Exit" (see Figure 5.1) at the top. For the experiments presented in this thesis we use a 16×16 cm stage (see Figure 5.1). The characters visualized in the Dead End game (as illustrated in Figure 5.1) are a dark grey circle of radius 0.75 cm representing the player, named '*Cat*', and a number of light grey square (of dimension 1.5 cm) characters representing the opponents, named '*Dogs*'. The aim of the *Cat*, starting from a randomly chosen position at the bottom of the stage, is to reach the Exit by avoiding the *Dogs* or to survive for a predetermined large period of time, i.e. 50 simulation steps. On the other hand, *Dogs* are aiming to defend the Exit and/or catch the *Cat* within that period of time. The name 'Dead End' is devised to demonstrate the situation in which the *Cat* finds itself at the beginning of each game. The game's fundamental concepts are inspired by previous work of Yannakakis et al. (2003) while the first use of the game as a test-bed for experiments on emergent cooperative opponent behaviors is introduced in (Park, 2003).

Since, conceptually, there are several *Dogs* on the game field, they are designed to be slower than the *Cat* so that the game is fairer to play. The *Cat* moves at four thirds the *Dogs'* maximum speed and since there are no dead ends, it is impossible for a single *Dog* to complete the task of killing it. Given that the *Cat* is faster than a *Dog*, the only effective way to kill the *Cat* is for a group of *Dogs* to hunt cooperatively. It is worth mentioning that one of the *Dogs'* properties is permeability: two or more *Dogs* can simultaneously occupy the same position on the game field.

Cat and *Dogs* are initially placed in the game field so that there is a suitably large

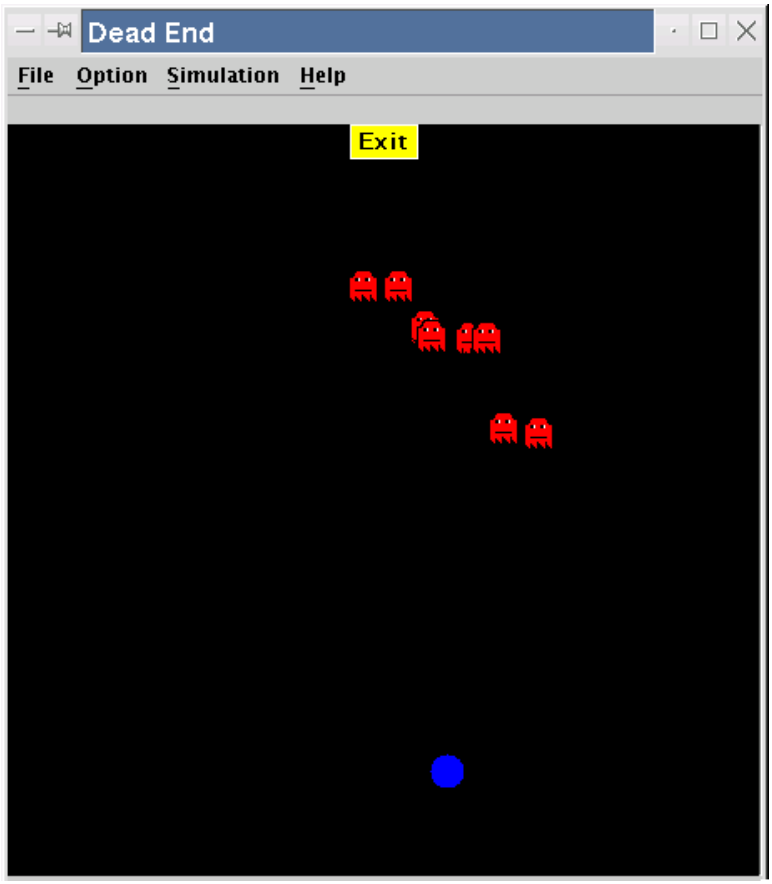


Figure 5.1: A snapshot of the Dead End game.

distance between them. Then, the following occur at each simulation step of the game:

1. Both *Cat* and *Dogs* gather information from their environment and take a movement decision, up, down, left or right.
2. If the game is over (i.e. *Cat* escapes through the Exit, *Cat* is killed, or the simulation step is greater than 50), then a new game starts from the same initial positions for the *Dogs* but from a different, randomly chosen position, at the bottom of the stage for the *Cat*.

5.1.1 Cat

The difficulty of the Dead End game is directly affected by the intelligence of the *Cat*. Its nature is significant because *Dogs'* emergent behavior is strongly related to their competitive relationship against it. To develop more diverse agents' behaviors, different playing strategies are required. We therefore chose three fixed *Dog*-avoidance

and/or Exit-achieving strategies for the *Cat*, differing in complexity and effectiveness. Each strategy is based on decision making applying a cost or probability approximation to the *Cat*'s four directions.

As previously mentioned, the *Cat* starts a game at a random position at the bottom of the game field and its aim is to reach the game's Exit by avoiding the *Dogs*. The non-deterministic initial position is devised to provide *Dogs* with diverse examples of playing behaviors to learn from.

5.1.1.1 Randomly-Moving (RM) *Cat*

The RM *Cat* takes a movement decision by selecting a uniformly distributed random picked available (no wall) direction at each simulation step of the game. The probability of selecting the direction towards the Exit linearly increases over the simulation steps by 0.2% per step.

5.1.1.2 Exit-Achieving (EA) *Cat*

The EA *Cat* moves directly towards the Exit. Its strategy is based on moving so as to reduce the greatest of its relative coordinates from the Exit.

5.1.1.3 Potential Field-Based (PFB) *Cat*

This constitutes the most efficient *Dog*-avoiding and Exit-achieving strategy of the three different fixed-strategy *Cat* types. A discrete Artificial Potential Field (APF) (Khatib, 1986), specially designed for the Dead End game, controls the PFB *Cat*'s motion (see also Section 4.5.2 for details on the Animals' APF in the *FlatLand* world). The essence of the APF for the Dead End game is that points along the *Cat*'s path to the Exit are considered to be attractive (i.e. low moving cost points), while obstacles (i.e. *Dogs*) in the environment are repulsive (i.e. high moving cost points). The overall APF causes a net force to act on the *Cat*, which guides it along a *Dog*-avoidance, Exit-achievement path. For illustration, consider the PFB *Cat* as a small cube that slides down the surface illustrated in Figure 5.2.

This surface is plotted by the PFB *Cat* at every simulation step and represents the function $C(x,y)$:

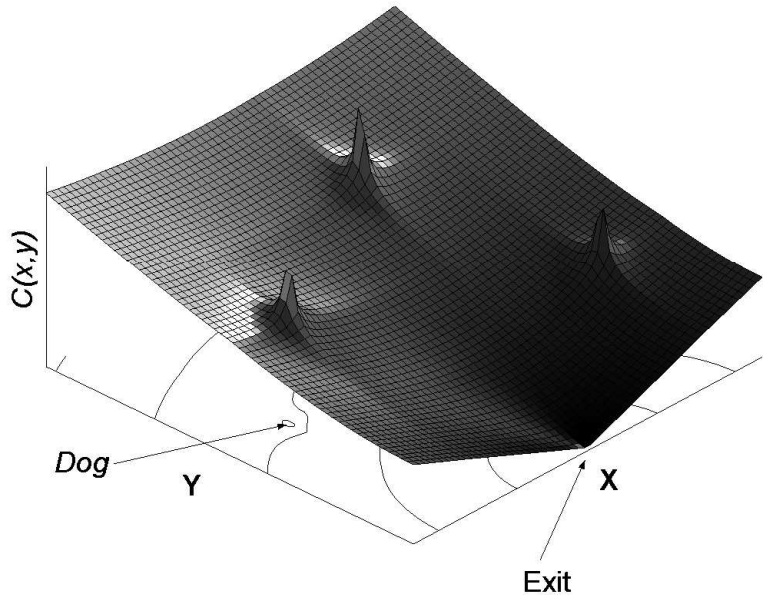


Figure 5.2: APF of the PFB *Cat*; situation of three obstacles — *Dogs* $(\mathrm{N} = 3)$.

$$C(x,y) = \Delta_E(x,y) + D(x,y) \tag{5.1}$$

$$\Delta_E(x,y) = \sqrt{(x_e - x)^2 + (y_e - y)^2} \tag{5.2}$$

$$D(x,y) = \sum_{i=1}^{\mathrm{N}} \frac{\rho}{|x_{d,i} - x| + |y_{d,i} - y|} \tag{5.3}$$

where $C(x,y)$ is the cost of the grid square (x,y); N is the total number of *Dogs* in the game field; (x_e, y_e) are the cartesian coordinates of the Exit's center; $(x_{d,i}, y_{d,i})$ are the current cartesian coordinates of the i^{th} *Dog's* center; ρ is a parameter that defines the height of the *Dog's* cost 'hill' function presented in (5.3) — for the experiments presented in this thesis $\rho = 1000$ (note that the PFB *Cat* can 'see' all the *Dogs* while a *Dog* can only 'see' a selected number of its nearest neighbors — see Section 6.1.3.1).

A PFB *Cat*, at each simulation step, calculates the moving cost (see (5.1)) of each grid square in a circle of radius 2 cm within the game field, centered at its current position. Then, the PFB *Cat* moves 2 cm to the grid square of minimal cost on the perimeter of the circle by following the grid-based trajectory of minimal average cost. While, in theory, APFs may be prone to local minima, in practice, in the dynamic Dead End

game, the probability for such cases to occur is significantly low and, therefore, can be ignored.

Any motion strategy that guides a *Cat* to arrive quickly at the Exit, avoiding any *Dogs* and keeping to the straightest and fastest possible trajectory, is definitely a "good" strategy in terms of the Dead End game. Hence, the PFB *Cat* presents a "good" behavior in our computer game and furthermore a reference case to compare to human playing behavior.

5.1.2 Neural Controlled Dogs

A feedforward neural controller is employed to manage the *Dogs'* motion and is described here.

5.1.2.1 Input

Using their sensors, *Dogs* inspect the environment from their own point of view and decide their next action. Each *Dog* receives input information from its environment expressed in the neural network's input array of dimension $(2\mathbf{z}+4)$ — see Figure 5.3. The input array consists of the relative coordinates of (a) the *Cat* in x ($\Delta_{x,P} = x_d - x_p$) and y ($\Delta_{y,P} = y_d - y_p$) axis, (b) the $\mathbf{z}$ closest *Dogs* in x ($\Delta_{x,C} = x_d - x_c$) and y ($\Delta_{y,C} = y_d - y_c$) axis and (c) the Exit in x ($\Delta_{x,E} = x_d - x_e$) and y ($\Delta_{y,E} = y_d - y_e$) axis; where (x_d, y_d), (x_p, y_p), (x_e, y_e) and (x_c, y_c) are the cartesian coordinates of the current *Dog's*, the *Cat*'s, the Exit's and the closest *Dogs'* current positions respectively.

All input values are linearly normalized into [-1, 1] via $\Delta_{i,J}/L_i$ where $i \in \{x,y\}$, $J \in \{P,C,E\}$ and L_x, L_y are the width and height of the stage respectively.

5.1.2.2 Architecture

As previously mentioned, a multi-layered fully connected feedforward neural network has been used for the experiments presented here (see Figure 5.4). The hyperbolic tangent sigmoid function is employed at each neuron.

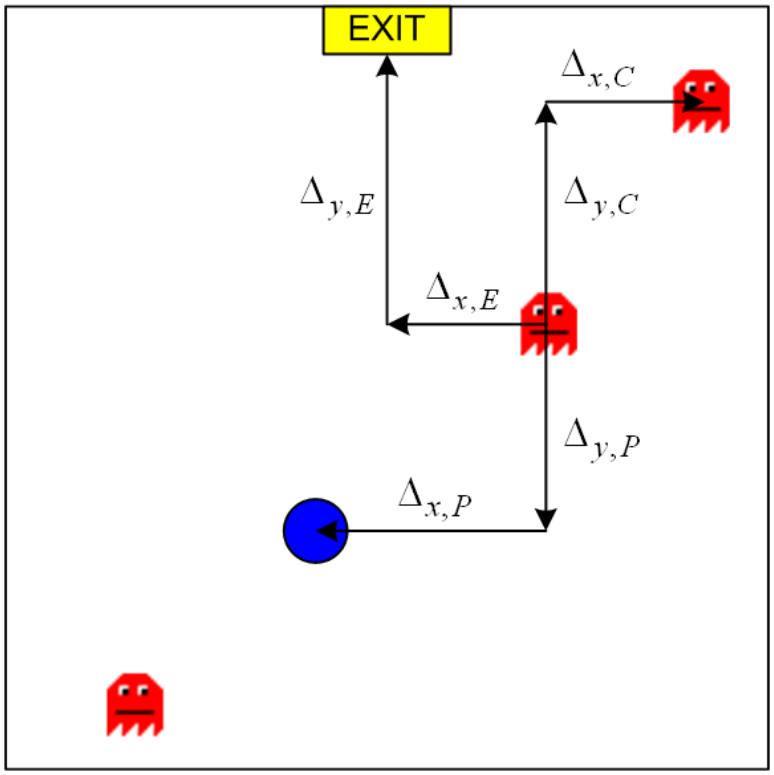


Figure 5.3: *Dog's* environment perception — Case of one closest neighbor ($\mathbf{z} = 1$).

5.1.2.3 Output

The neural network's output is a two-dimensional vector $[o_1, o_2]$ with respective values from -1 to 1. This vector represents the *Dog's* chosen motion and is converted into cartesian coordinates according to (5.4) and (5.5).

$$x_d^{k+1} = \begin{cases} x_d^k, & \text{if } |o_1| \geq |o_2| \quad (5.4a) \\ x_d^k + o_2 s, & \text{if } |o_1| < |o_2| \quad (5.4b) \end{cases}$$

$$y_d^{k+1} = \begin{cases} y_d^k + o_1 s, & \text{if } |o_1| \geq |o_2| \quad (5.5a) \\ y_d^k, & \text{if } |o_1| < |o_2| \quad (5.5b) \end{cases}$$

where (x_d^k, y_d^k) are the cartesian coordinates of the *Dog's* center at simulation step k; s is the *Dog's* maximum speed — for the experiments presented here $s = 1.5$ cm/simulation step (this being 3/4 of the *Cat*'s speed).

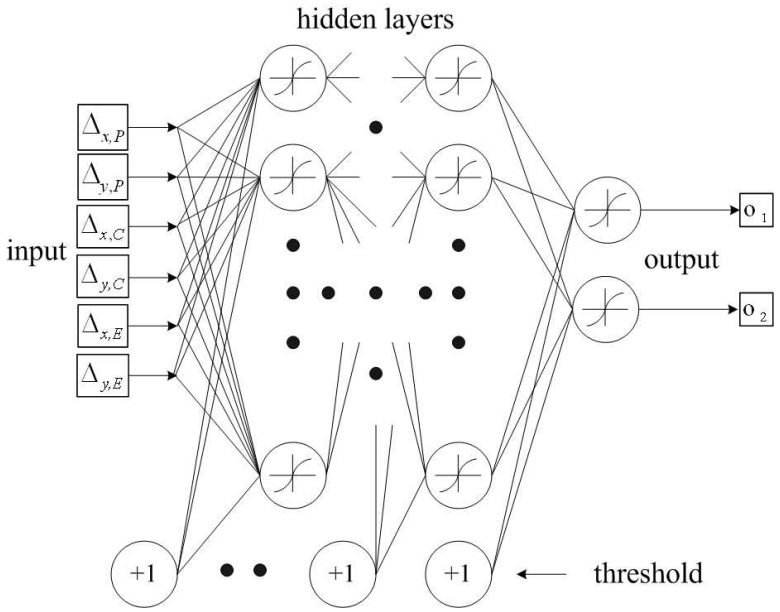


Figure 5.4: Multi-layered fully connected feedforward neural network controller for the *Dog*— Case of one closest neighbor ($\mathbf{z} = 1$).

5.1.3 Fixed Strategy Dogs

Apart from the neural controlled *Dogs*, an additional fixed non-evolving strategy has been tested for controlling the *Dogs'* motion. *Dogs* of this strategy are called 'Followers' and they are designed to follow the *Cat* constantly by moving at their maximum speed (i.e. 1.5 cm/simulation step). Their strategy is based on moving so as to reduce the greatest of their relative coordinates ($\Delta_{x,P}, \Delta_{y,P}$) from the *Cat*. This strategy is used as a baseline behavior for comparison with any emergent neural controller behavior.

5.2 Challenges

Dead End is a hard environment for an agent to achieve behaviors of high performance because of the following distinct features (see also challenges for *FlatLand*'s Humans in Section 4.6):

- It is a fully dynamic multi-agent environment, in which each *Dog* moves continuously in the game field while interacting with a number of other *Dogs* and the *Cat*.
- In the experiments presented here, no agent has full information: *Dogs* can 'see'

the *Cat* and at most **z** other *Dogs* while the advanced PFB *Cat* can see all the *Dogs'* positions but not their future movements.

- Communication between the *Dogs* is limited to each being able to 'see' the position of **z** nearest neighbor *Dogs* — cooperative action must be built on this implicit and partial communication.
- A *Dog's* input is discontinuous because its nearest neighbor(s) alter; hence the values of the relative coordinates $(\Delta_{x,C}, \Delta_{y,C})$ also change, in a discontinuous fashion.

The basic concept and features of Dead End make it interesting to the multi-agent predator/prey computer games field. Its key features are that cooperative behavior amongst the *Dogs* is necessary and is supported only by implicit partial communication and the on-line learning mechanism that we propose (see Section 5.6) allows the *Dogs* constantly to adapt their collective strategies as they interact with the *Cat*, contributing to the interest of the game.

5.3 Interest Parameter Values

In this section we present the procedures followed to obtain the appropriate parameter values of the interest estimate (2.5) for the Dead End game.

5.3.1 Minimum Playing Time — t_{min}

For the experiments presented here, t_{min}, which is an estimate for the minimum playing time, is 3 simulation steps. This is obtained as the minimum simulation time recorded that any *Cat* type survives when playing against any opponent *Dogs*.

5.3.2 Maximum Playing Time — t_{max}

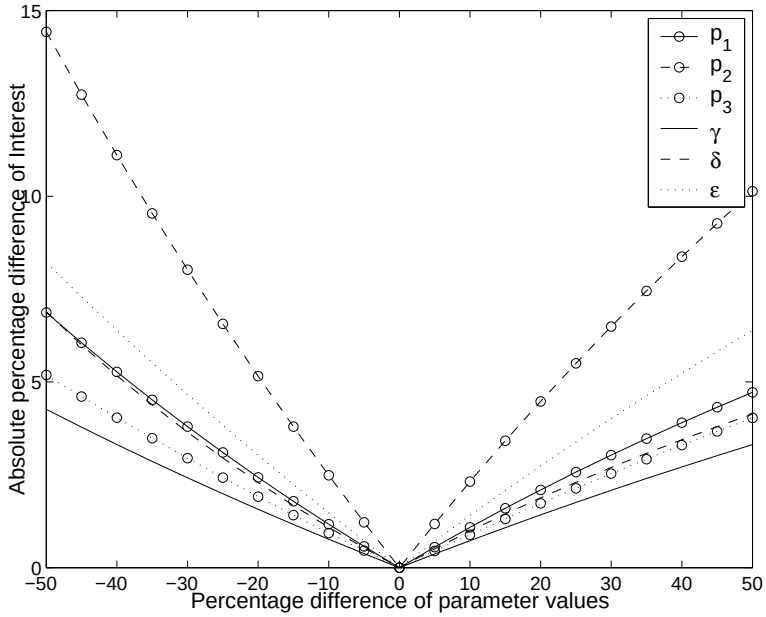
As previously defined, t_{max} is the maximum evaluation period of play, or else the maximum lifetime of the player. In the game of Dead End t_{max} is determined by the maximum game length recorded against any opponent *Dogs* and it is 50 simulation steps for the RM and PFB *Cat* and 10 simulation steps for the EA *Cat*.

5.3.3 Weighting Parameters

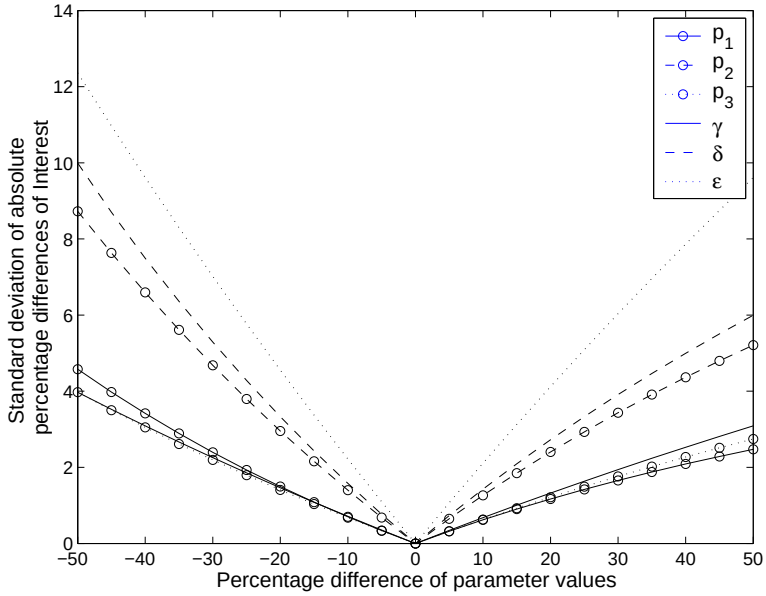
In order to obtain values for the interest criteria weighting parameters γ, δ and ε we select empirical values based on the specific game. For Dead End, diversity in game play is of the greatest interest. We believe that generating diverse behaviors within this game should be weighted more than the challenge (T) and the spatial diversity ($E\{H_n\}$) criteria since the game period is small, that is, the game is short on average. Given the above-mentioned statements and by adjusting these three parameters so that the interest value escalates as the opponent behavior changes from Random to Follower, we come up with $\gamma = 1, \delta = 2$ and $\varepsilon = 1$.

Since the interest value changes monotonically with respect to each of the three criterion values T, S, $E\{H_n\}$, sensitivity analysis is conducted on all interest metric parameters aiming to portray the relation between these parameters as well as their weighting degree in the interest formula. We therefore proceed by seeking opponent behaviors that generate ten different T, S and $E\{H_n\}$ values, equally spread in the [0,1] interval. Given these thirty values as input, p_1, p_2, p_3 ($p_1 = 0.5$, $p_2 = 1$ and $p_3 = 4$ — see Chapter 2), γ, δ and ε parameters are systematically changed one at a time so that their percentage difference lies in the interval $[-50\%, 50\%]$. We believe that fifty is a large enough percentage difference to demonstrate potential significant impact on the observed value. Each time a parameter change occurs, the absolute percentage difference of the game's interest is computed. The function between the absolute percentage differences of the interest value and the percentage differences of the interest weighting parameters is illustrated in Figure 5.5(a) and Figure 5.5(b).

The first conclusion that arises from Figure 5.5(a) is that changes on the p_1 and p_2 parameters seem to affect the I value more than their respective criterion weights γ, δ. The observed difference in interest sensitivity is reasonable since the first two parameters represent powers while the latter three correspond to product weights. More specifically, p_1 and p_2 reveal significant differences (i.e. greater than 5%) in I respectively when decreased by 38% (i.e. $p_1 = 0.31$) and 18% (i.e. $p_2 = 0.82$) or when p_2 is increased by 20% (i.e. $p_2 = 1.2$). For p_3 significant change in I is observed only when decreased by up to 49% (i.e. $p_3 = 2.04$). Accordingly, both δ and ε demonstrate significant differences in I when decreased by 40% and 30% respectively. The ε parameter does also significantly change the I value when it is increased by 35%. Finally, for γ no significant change in I is observed even when changed by up to 50%.



(a) Average



(b) Standard deviation

Figure 5.5: Average and standard deviation of absolute percentage differences of I over ten runs for each weighting parameter.

Regardless of the sensitivity of the I value, mainly as far as the p_1 and p_2 parameters are concerned, we believe that the selected values project a rather robust I value

considering the fact that they constitute power parameters in the interest formula.

5.4 Performance Measurement

We introduce an efficient method for testing and comparing different learning attempts' ability to emerge successful controllers. We record the total number of kills (*Cat* is killed) K of the examined team of *Dogs*, against a specific *Cat*, by placing these agents in Dead End and letting them play 100 games, since we believe it is a long enough period for testing a playing-behavior of a team of *Dogs* in an efficient way. This evaluation is called a *trial*. The performance P of this group of *Dogs* equals to the number of *Cat*-kills K (i.e. $P = K$).

5.5 Off-Line Learning

We use an off-line evolutionary learning approach in order to produce some 'good' (i.e. high performing) and diverse initial behaviors for the on-line learning mechanism. The ANNs that determine the behavior of the *Dogs* are evolved (evolutionary process is limited to the connection weights of the ANN).

The evolutionary procedure we follow is based on the GGA algorithm presented in Section 4.1.1.1, being the most robust and effective off-line learning approach in the prototype *FlatLand* world. Each *Dog* has a genome that encodes the connection weights of its ANN. A population of 40 ANNs (*Dogs*) is initialized randomly with initial uniformly distributed random connection weights that lie within [-5, 5]. Then, the off-line learning algorithm follows the GGA approach with specific adjustments for the Dead End game only in its first two basic steps:

Step 1: Each *Dog*'s clones are placed in the Dead End game field and play the game against a selected *Cat* type for an evaluation period e_p (e.g. 125 simulation steps). The outcome of this game is to ascertain the total number of kills (K) and wins (W) in the number of finished games G within the e_p period ($G = W + K$).

Step 2: Each *Dog* is evaluated via (5.6)

$$f = r_K K - r_W W \tag{5.6}$$

where r_K is the reward rate of a kill; r_W is the penalty rate of a win. By using (5.6), we promote *Dogs* (their N clones) that are able to kill the *Cat* as many times as possible as well as to defend the Exit successfully during an evaluation period. We expect that by adjusting r_K and r_W, *Dogs* of different behaviors will emerge.

For the experiments presented here $N_s = 20\%$ and $p_m = 0.02$. The algorithm is terminated when a predetermined number of generations $\mathbf{T}$ is achieved (e.g. $\mathbf{T} = 300$) and the best-fit *Dog's* connection weights are saved. Opponents trained through this procedure are said to be off-line trained (OLT).

Dogs play for a small period (i.e. $e_p = 125$ simulation steps) when evaluated by the off-line learning mechanism. This evaluation procedure constitutes an approximation of the examined *Dogs*' overall performance in larger evaluation periods and keeps the computational cost low.

5.6 On-Line Learning (OLL)

This learning approach is based on the idea of opponents that learn while they are playing against the *Cat*. In other words, *Dogs* that are reactive to any *Cat*'s behavior and learn from its strategy instead of being predictable and, therefore, uninteresting characters for game-playing. Furthermore, this approach's additional objective is to keep the game's interest at high levels as long as it is being played. The OLL mechanism is built upon the algorithm presented in Section 4.3.

Beginning from any initial group of homogeneous off-line trained *Dogs*, the OLL mechanism transforms them into a group of heterogeneous characters that are conceptually more interesting to play against. In OLL, an OLT opponent is cloned a number of times, that equals to the number of opponents playing the game, and its clones are placed in the game field to play against a selected fixed player type. Then, the OLL approach follows the algorithm presented in Section 4.3 with adjustments for the specific game only in the following steps.

Step 1 Each *Dog* is evaluated every e_p simulation steps via (5.7), while the game is played — e_p is 25 simulation steps.

$$f_{OLL} = \sum_{i=1}^{e_p} \left\{ D_{P,i} - D'_{P,i} \right\} \tag{5.7}$$

where

$$D_{P,i} = |x_d^{i+1} - x_p^i| + |y_d^{i+1} - y_p^i| \quad (5.8)$$

$$D'_{P,i} = |x_d^i - x_p^i| + |y_d^i - y_p^i| \quad (5.9)$$

and (x_d^i, y_d^i), (x_p^i, y_p^i) are respectively the cartesian coordinates of the *Dog* and the *Cat* at the i^{th} simulation step. This fitness function promotes opponents that move towards the player within an evaluation period of e_p simulation steps.

Step 2 A pure elitism selection method is used where only the fittest solution is able to breed. The fittest parent clones an offspring with a probability p_c that is inversely proportional to the normalized *Dogs* cell visit entropy (i.e. $p_c = 1 - H_n$). If there is no cloning, then go back to Step 1, else continue to Step 3.

Step 3 Mutation occurs in each gene (connection weight) of each offspring's genome with a probability p_m (e.g. 0.02). A uniform random distribution is used to define the mutated value of the connection weight. The mutated offspring replaces the least-fit member of the population and takes its position in the game field — this is also called 'replacement method' throughout the dissertation. Note that, for the Dead End game, the mutated offspring is not evaluated briefly in off-line mode as described in Section 4.3 (Step 4 of the algorithm). See Chapter 6 for further discussion on this omission.

The algorithm is terminated when a predetermined number of generations has been achieved or a game of high interest is found. Figure 5.6 illustrates the main steps of the on-line learning algorithm.

We mainly use small simulation periods (i.e. $e_p = 25$) to evaluate *Dogs* in on-line learning. The aim of this high frequency of evaluations is to accelerate the on-line evolutionary process without significantly affecting the real-time performance of the game.

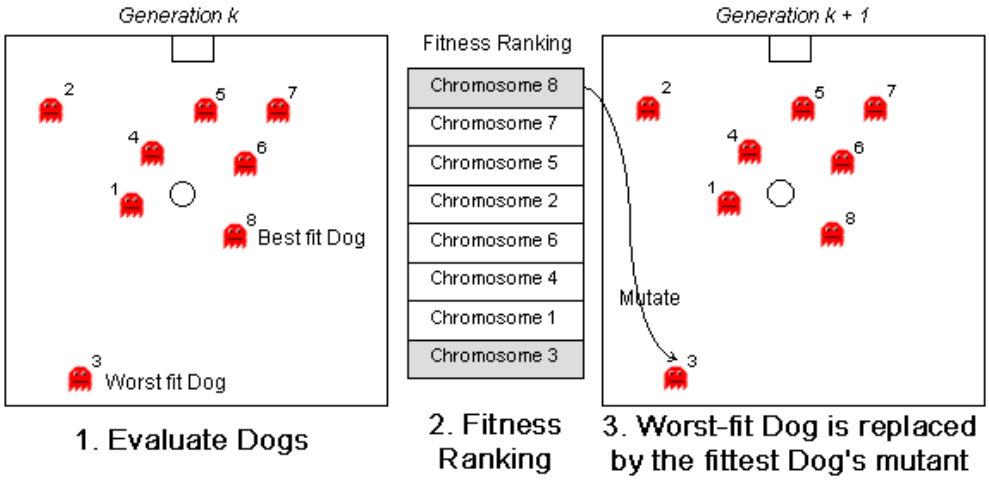


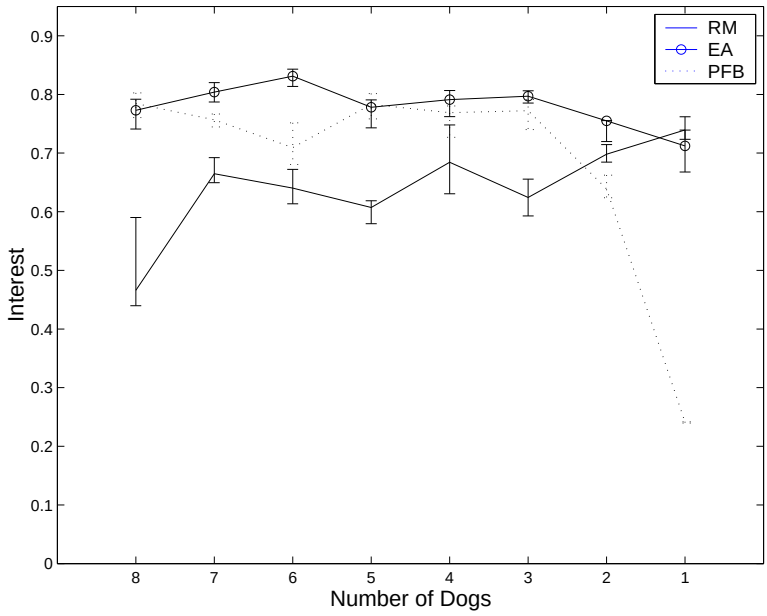
Figure 5.6: The OLL mechanism with the replacement method.

5.7 Game Features

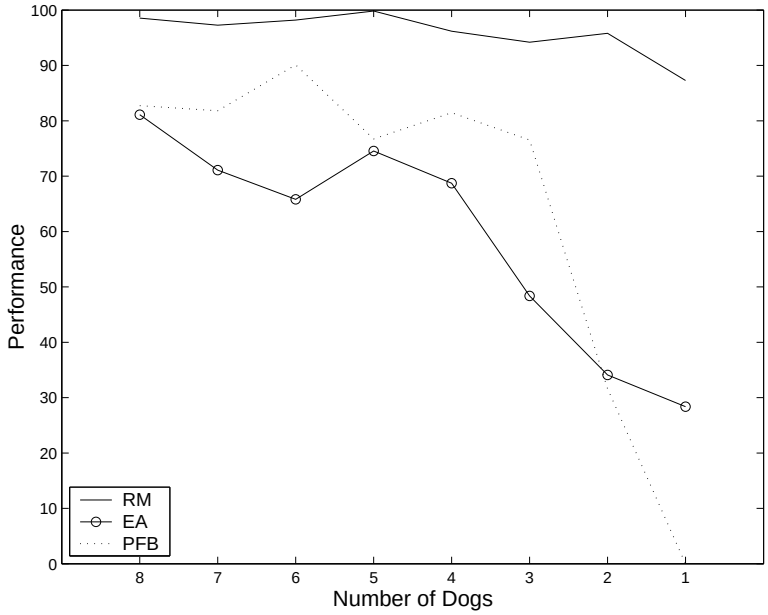
5.7.1 Number of Dogs

Since the game's complexity is directly affected by the number of opponents, a study needs to be carried out in order to determine the boundaries between which the game is neither too hard nor too easy to play. The playing strategy of the Followers is an effective way to measure the difficulty of the game given the fixed-strategy hand-programmed *Cats*. On that basis, we let a number of Followers, varying from 8 to 1, play 100 games against each of the fixed-strategy *Cat* types and we calculate their interest (through the bootstrapping procedure presented in Appendix A— $N = 50$) and performance values which appear in Figure 5.7(a) and Figure 5.7(b) respectively.

Figure 5.7(b) suggests that the game becomes too challenging for a potential human player when there are more than 5 *Dogs* on the stage. Likewise, the game becomes relatively easy when there are less than 3 opponents in the game. We reach this conclusion primarily by observing the Followers' performance values against all *Cat* types, which as a game scenario simulates various indicative average-skilled human playing behaviors against some well-behaved and effective opponents. In addition, the aforementioned number of Followers seems to generate the highest interest values among all environments tested. Consequently, for the experiments presented in this chapter, we will explore Dead End environments consisting of 3, 4 and 5 *Dogs* (except where otherwise noted).



(a) Interest



(b) Performance

Figure 5.7: Followers' interest and performance values over the number of *Dogs* in Dead End .

5.7.2 Sensory Information

One of the primary aims of this work is to focus on the minimal sensing information capable of generating collaborative opponent behaviors. In order to determine

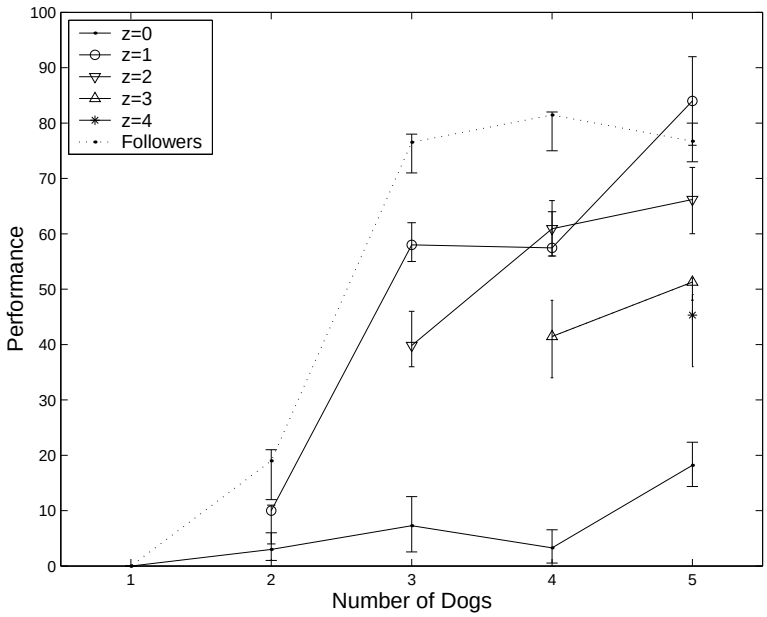


Figure 5.8: *Dogs*' performance average and interval values over 10 off-line learning attempts against the PFB *Cat*.

the appropriate amount of sensory information (given the above-mentioned goal) the following experiment is devised. We train *Dogs* off-line (see Section 5.5) against the PFB *Cat*, being the most advanced of the three *Cat* types. In this experiment we select $r_K = r_W = 1$ in fitness function (5.6) — providing equal opportunities for promoting both *Cat*-hunting and Exit-defensive behaviors. For each of the 2, 3, 4 and 5 *Dog* game environments we explore all possible sensing scenarios; starting from the minimal sensing scenario ($\mathbf{z} = 0$) to global positioning perception, that is each *Dog* 'sees' the positions of all the other *Dogs* appearing in the environment. We repeat the learning attempt (run) 10 times with different initial conditions. The experiments' obtained performance and its confidence intervals are displayed in Figure 5.8.

The primary deduction that arises from Figure 5.8 is that information about neighbor agents helps towards cooperative behaviors which yield high performance values. In particular, *Dogs* are able to cooperate when even the position of only one closest neighbor *Dog* is perceived. This suggest that one neighbor *Dog* constitutes the minimal information for emerging cooperative behaviors. Because we are interested specifically in the minimal sensing scenario (for computational effort purposes), we will deliberately exclude from consideration any global sensing, e.g. information about the dispersion of the *Dogs* as a whole. Experiments conducted in Dead End are built on this scenario (i.e. $\mathbf{z} = 1$).

5.7.3 Controller Size

The modified 1-hidden layer ECWAS algorithm was used and the experiment presented in Section 4.8.2.1 was conducted for obtaining a neural-controller of minimal size for the *Dogs*. As in the *FlatLand* prototype, the 5-hidden neuron architecture appeared to be the most frequent highly efficient structure designed by ECWAS for the Dead End game. Experiments were held on the 3-opponent game against the PFB *Cat* and the 5-hidden neuron ANN was automatically designed in 8 out of 40 ECWAS trials with an average performance of 61.13. This controller will be used for all Dead End experiments.

5.8 Off-Line Learning Experiments

The experiment presented here is focused on producing well-behaved *Dogs* in terms of the performance measure previously described in Section 5.4. We train *Dogs* against all three fixed-strategy *Cat* types through the off-line learning mechanism ($r_K = r_W = 1$). The off-line learning experiment is described as follows.

For each of the 3, 4 and 5 *Dogs* game environments: (a) apply the off-line learning mechanism by playing against each *Cat* type separately. Repeat the learning attempt (run) 10 times with different initial conditions. (b) Evaluate each of the 10 teams of OLT Dogs against all three *Cat* types. Their performance and interest measurements are given by the average values obtained over the 10 trials. (c) Evaluate non-evolving randomly generated (i.e. untrained) *Dogs* and Followers against every *Cat* type (run 10 trials and calculate their average performance and interest). The outcome of this experiment is presented in Table 5.1 and Table 5.2.

According to Table 5.1, in most cases, OLT *Dogs* against a specific *Cat* seem to achieve lower average performance values when rivaling a *Cat* other than the one they have been trained against off-line. This is the case in OLT *Dogs* against the RM *Cat* (noted as OLT/RM in Table 5.1) which produces bad generalizations against the other two playing strategies. OLT *Dogs* against the EA *Cat* manage to perform well when playing against the RM *Cat*; however, they perform poorly when playing against the PFB *Cat*.

On the other hand, *Dogs* trained off-line against the PFB *Cat* show good overall performance against all *Cat* types. Even though OLT/PFB *Dogs* do not achieve high average

		Playing against						
		RM		EA		PFB		Mean
	Dogs	$E\{P\}$	σ^2	$E\{P\}$	σ^2	$E\{P\}$	σ^2	$E\{P\}$
OLT/RM	5	94.5	0.22	44.3	3.63	38.1	18.12	52.84
	4	94.2	3.40	34.9	7.54	26.2	36.21	
	3	92.8	4.82	28.5	13.76	19.8	48.45	
OLT/EA	5	62.6	9.33	96.1	1.15	49.3	15.93	64.56
	4	70.9	8.27	73.6	4.59	43.8	23.05	
	3	81.1	2.61	69.4	6.67	34.3	28.23	
OLT/PFB	5	95.1	2.22	52.5	17.48	84.0	5.10	67.82
	4	90.4	14.31	45.8	31.71	57.5	11.35	
	3	87.0	21.89	40.1	21.08	58.0	18.56	
Followers	5	99.8	0.13	74.5	0.65	76.7	5.70	
	4	96.2	0.65	68.7	5.99	81.5	7.78	
	3	94.2	3.54	48.4	21.77	76.5	5.09	
Untrained	5	75.6	163.98	62.5	214.58	17.8	288.03	
	4	40.2	198.25	15.7	318.35	0.00	0.0	
	3	31.2	195.15	11.8	379.45	0.00	0.0	
Mean of $E\{P\}$		80.38		51.12		44.23		

Table 5.1: The effect of off-line training on the *Dogs*' average performance $(E\{P\})$ values over 10 learning attempts.

performance values when playing against the EA *Cat* they achieve the highest mean of the average performance values against all players (i.e. the mean of the $E\{P\}$ values on each OLT behavior row of Table 5.1). Therefore, among the three fixed-strategy players, the PFB *Cat* provides the best off-line training for the opponent agents. This suggests that when *Dogs* learn from more complex and effective types of players, they tend to generalize better.

Performance results obtained from off-line learning experiments also demonstrate the difference in effectiveness of the fixed playing strategies used. It is obvious that the RM *Cat* (mean performance over all off-line training attempts equals to 80.38; presented in the bottom row of Table 5.1) is the least effective and 'easiest to kill' player, whereas, the EA (51.12) is harder to kill and the PFB (44.23) proves to be the most effective playing strategy of all three.

		Playing against						
		RM		EA		PFB		Mean
	Dogs	$E\{I\}$	σ	$E\{I\}$	σ	$E\{I\}$	σ	$E\{I\}$
OLT/RM	5	0.696	0.0450	0.754	0.0107	0.518	0.0068	0.501
	4	0.708	0.0104	0.724	0.0086	0.619	0.0084	
	3	0.702	0.0227	0.683	0.0168	0.606	0.0317	
OLT/EA	5	0.555	0.0172	0.661	0.0242	0.486	0.0068	0.404
	4	0.488	0.0094	0.582	0.0075	0.424	0.0161	
	3	0.549	0.0117	0.584	0.0038	0.519	0.0037	
OLT/PFB	5	0.625	0.0274	0.716	0.0129	0.493	0.0174	0.467
	4	0.627	0.0270	0.648	0.0281	0.615	0.0158	
	3	0.656	0.0116	0.624	0.0337	0.599	0.0573	
Followers	5	0.607	0.0804	0.778	0.0256	0.783	0.0211	
	4	0.684	0.0588	0.791	0.0226	0.768	0.0281	
	3	0.624	0.0314	0.797	0.0105	0.772	0.0259	
Untrained	5	0.491	0.0190	0.498	0.0572	0.425	0.0058	
	4	0.614	0.0316	0.436	0.0496	0.056	0.0120	
	3	0.561	0.0255	0.312	0.0140	0.236	0.0022	
Mean of $E\{I\}$		0.612		0.638		0.527		

Table 5.2: The effect of off-line training on the *Dogs*' average interest $(E\{I\})$ values over 10 learning attempts.

An increased interest value when *Dogs* are trained off-line is also noticeable in the majority of cases (compared to the interest generated by the untrained *Dogs* — see Table 5.2). However, these emergent behaviors fail to compete with the interest generated by the Followers (mainly against the EA and PFB *Cat*).

Table 5.2 also demonstrates that the *Cat* type may have an impact on the generated interest. It appears that the EA *Cat* is the most interesting *Cat* to play against followed by the RM *Cat* and the PFB *Cat*. This observation is consistent with the assumption about the quality of the player which is discussed in Chapter 2. According to this assumption, extreme-behaved players may not generate good estimates of the game's interest value. Both EA and (in a lesser degree) RM types of *Cat* belong to this category of game-playing by following a trivial strategy of low-quality. Further discussion on the limitations that arise from this assumption is presented in Chapter 9.

On the whole, the off-line learning mechanism generates *Dogs* that defend the Exit and/or hunt the *Cat* in a cooperative fashion. As noted before, opponents in this game have to learn to cooperate in order achieve a high performance value against any playing strategy. OLT-obtained behaviors are classified into the following two categories:

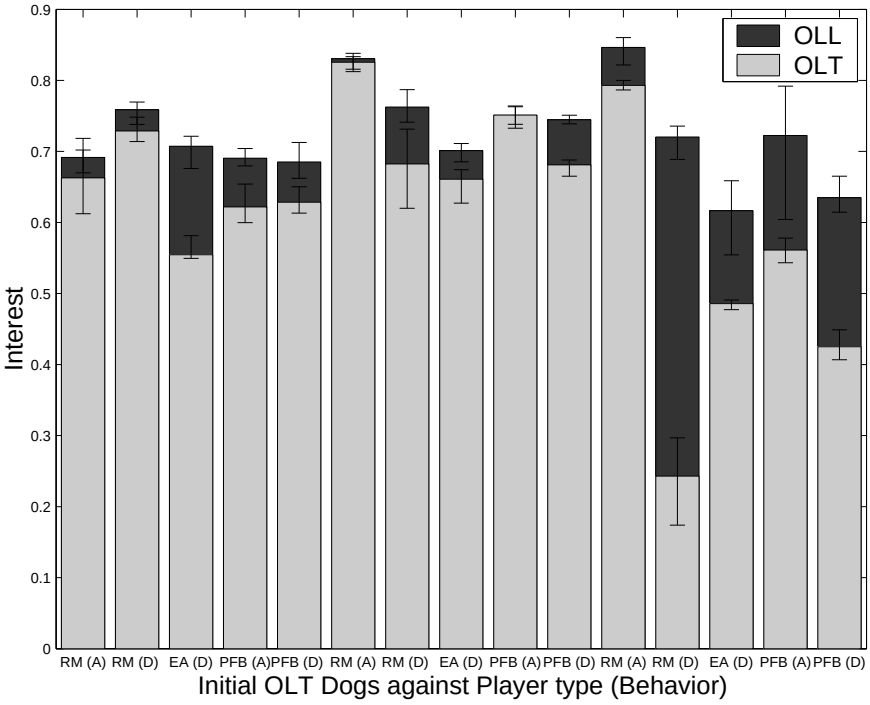
- Defensive (D): These are OLT *Dogs* that tend to flock close and around the Exit and wait for the *Cat* to approach in order to kill it. Their average normalized cell visit entropy value $E\{H_n\}$ is less than 0.7.
- Aggressive (A): These are OLT *Dogs* that tend to follow the *Cat* all over the stage in order to kill it ($E\{H_n\} \geq 0.7$).

Defending the Exit, as an emergent behavior, is much easier than hunting cooperatively and more effective when playing against the EA *Cat*. Thus, when off-line training occurs against the EA *Cat*, aggressive *Dog* behavior does not emerge because it constitutes a sub-optimal behavior.

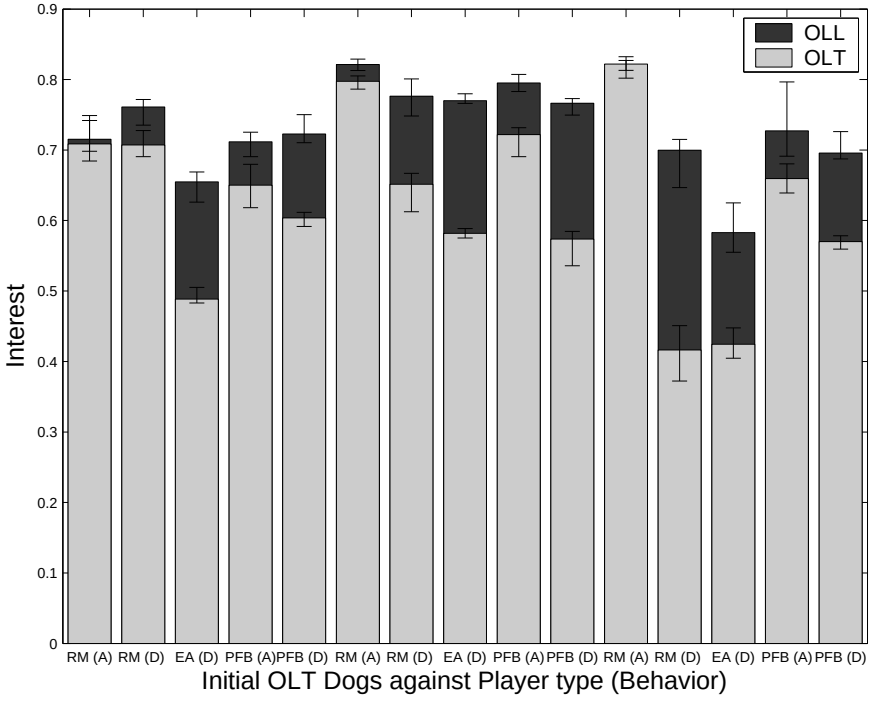
5.9 On-Line Learning Experiments

Some well-behaved *Dogs* are required initially to seed the OLL mechanism in its attempt to generate interesting Dead End games. Off-line trained emergent solutions serve this purpose. The OLL experiment for the Dead End game is described as follows. a) Pick five different emergent *Dogs'* behaviors produced from off-line learning experiments — Defensive (D) against each of the three *Cat* types and Aggressive (A) against the RM and the PFB *Cat* — for each of the three game environments — containing five, four and three *Dogs*; b) starting from each OLT behavior, apply the OLL mechanism by playing against each *Cat* type separately and in the same stage where off-line training occurred. This makes a total of fifteen different OLL attempts for each game environment. c) calculate the interest (by following the bootstrapping procedure presented in Appendix A — $N = 50$) of the game every 100 generations during each OLL attempt.

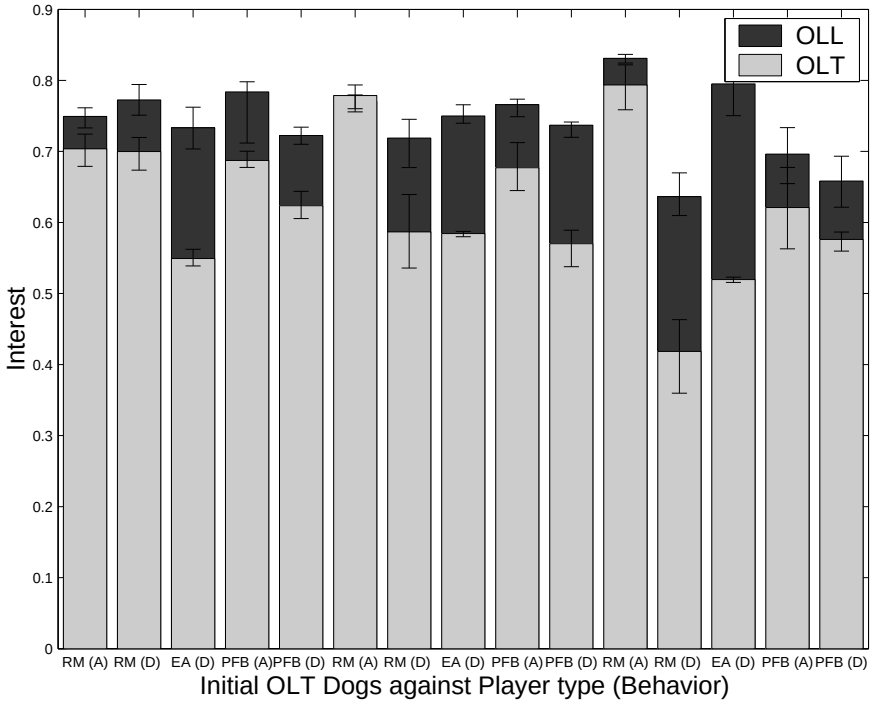
Given that there are three game environments explored, the total number of different OLL experiments is 45, which illustrate a complete picture of the mechanism's effectiveness over the three dimensions of the game: the game's complexity, the *Cat* type and the initial behavior (see Figure 5.9). The evolution of interest over the OLL gen-



(a) 5 Dogs



(b) 4 Dogs



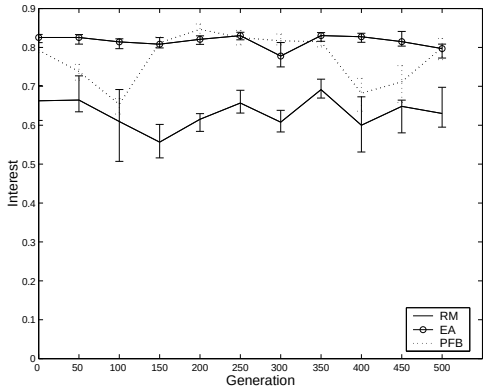
(c) 3 Dogs

Figure 5.9: (a), (b), (c): best interest values achieved from OLL in the three Dead End environments; Experiment Parameters: $e_p = 25$ simulation steps, $p_m = 0.02$, 5-hidden neurons controller.

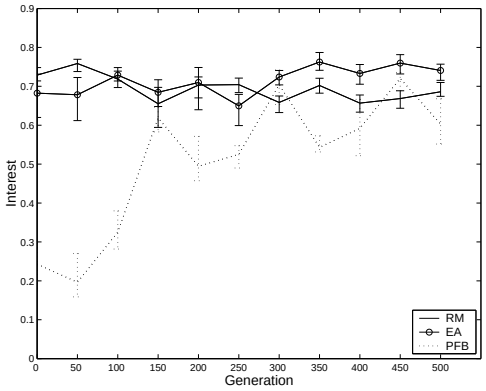
erations in the 5 *Dog* game environment are presented in detail in Figure 5.10 — see Appendix C for illustrations of the rest of the experiments.

As seen from Figure 5.9, OLL is successful at augmenting the interest of the game regardless of the stage complexity, the *Cat* type and the initial behavior. On that basis, in the majority of the experiments, OLL is capable of producing games of higher than the initial interest and/or maintaining that high interest for a long period. More comprehensively, in 42 out of 45 OLL scenarios the interest of the game is increased in less than 500 generations while in 37 cases this increase is statistically significant. Also, in 23 cases the best interest value achieved against a *Cat* type is greater than the respective interest value generated by the Followers (see Table 5.2). Given the calculated confidence intervals of the interest value (see Table C.1 in Appendix C), in 17 of such cases this difference is significant.

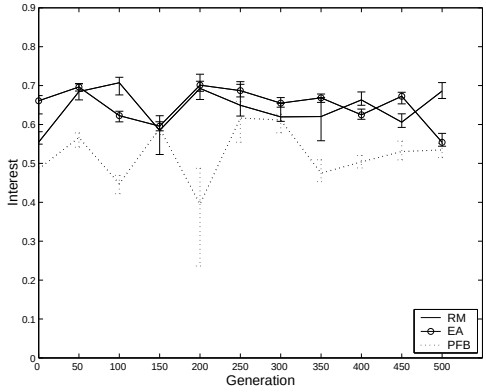
The reader might notice that the fewer the opponents are in the game, the more erratic



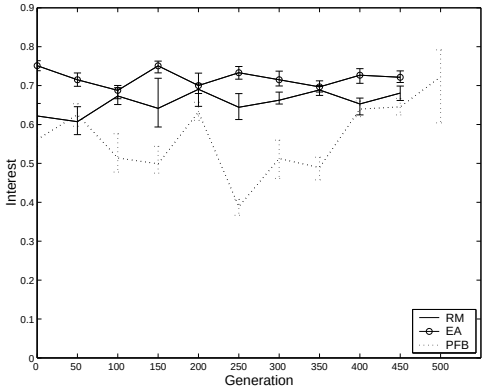
(a) RM Aggressive



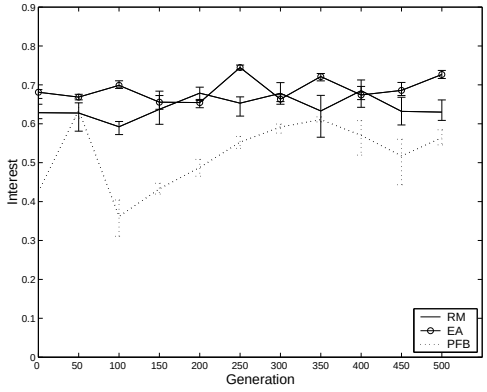
(b) RM Defensive



(c) EA Defensive



(d) PFB Aggressive



(e) PFB Defensive

Figure 5.10: Game interest over the number of OLL generations in the 5 *Dog* environment. Sub-figure captions denote the initial *Dogs*' behavior.

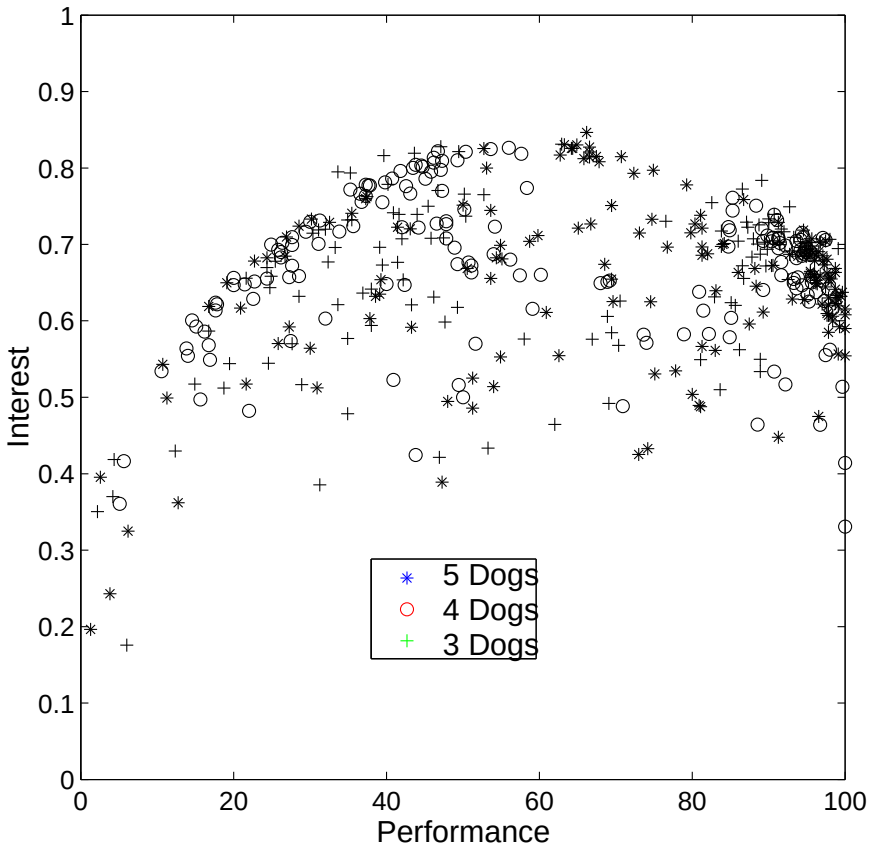


Figure 5.11: Scatter plot of I and P value instances for all three Dead End environments.

the behavior of the interest value is over the on-line learning games. This defines one of the challenges of learning in real-time, since the entertainment value (and performance value) contribution of a single *Dog* is inversely proportional to the size of the group of *Dogs*. Thus, bad or good opponent replacements in smaller groups are noticed easier by the player and may lead to the noisy behavior of the I value. However, as seen from Figure 5.10, Figure C.1 and Figure C.2, such erratic phenomena are unlikely to happen when the interest of the game is high (e.g. RM *Cat* in Figure 5.10(a)).

As already seen from both off-line and on-line learning experiments, behaviors of high performance ought to be sacrificed for the sake of highly entertaining games. Consequently, there has to be a compromise between P and I values as previously noted in Chapter 2. However, as seen from Figure 5.11, teamwork features within the *Dogs* behavior are maintained when interesting games emerge through the on-line learning mechanism. It appears that the most interesting games require a performance ($50 < P < 70$ approximately) which is not achievable without cooperation (see Figure 5.8). Thus, teamwork is present during on-line learning and it furthermore con-

tributes to the emergence of highly interesting games.

5.9.1 How Does OLL Work?

The fitness function (5.7) rewards 'aggressiveness' (movement toward the *Cat*); however, the OLL generated opponents become eventually more interesting. Herein, we will attempt to explain theoretically the correlation between aggression and interest that initially appears to be rather unexpected.

The OLL algorithm promotes mutation when groups of opponents exhibit low spatial diversity and, subsequently, the most aggressive *Dog* of the group has the opportunity to reproduce. This combination rewards both aggression explicitly and spatial diversity implicitly. Since estimating the interest value in real-time is an expensive procedure, aggression (via (5.7)) determines the interest value estimate that guides the on-line search towards more interesting opponents.

The primary reason why OLL is successful is because it is based conceptually on an active player-opponent interaction (see also the assumptions on the interest metric in Section 2.6). Hence, the more aggressive the opponents become through (5.7), the more challenging the game is for the player. Since, the player actively attempts to avoid them, it increases the spatial diversity of the opponents that are trying to follow him/her by uniformly covering the game environment. These behaviors collectively lead to the satisfaction of the challenge and the spatial diversity criteria in (2.5). Ultimately, the game reaches its highest interest when the player discovers new ways of playing that the opponents can cope with (increase of behavior diversity).

According to the interest metric assumptions presented in Section 2.6, a player that does not interact with his/her game opponents generates poor approximations of the entertainment value. Contrary to the PFB *Cat*, the EA and RM types of *Cat* do not interact with their opponents which may very well explain the significant differences on their generated I values. Even though humans (on average) are not expected to play blindly, such playing strategies reveal a limitation of the interest value estimation that is further discussed in Chapter 9.

5.10 Summary

The Dead End predator/prey computer game (Yannakakis *et al.*, 2004) is devised as an interesting test-bed for studying the emergence of multi-agent cooperative behaviors supported by partial and implicit communication through evolutionary learning mechanisms. We introduced an off-line learning mechanism, from which effective cooperative predator behaviors have rapidly emerged.

Predator strategies in predator/prey computer games are still nowadays based on simple rules, which even though they can generate highly complex opponent behaviors they make the game somewhat uninteresting — by the time the player gains more experience and playing skills. A computer game becomes interesting primarily when there is an on-line interaction between the player and his opponents who demonstrate adaptive behaviors.

Given some objective criteria for defining interest in predator/prey games we applied the method presented in Chapter 2 for explicitly measuring interest in the Dead End game. We saw that by using the proposed on-line learning mechanism (see also (Yannakakis and Hallam, 2004b)), maximization of the individual simple distance measure (see (5.7)) coincides with maximization of the game's interest. Apart from being robust, the proposed mechanism demonstrates fast adaptability to new types of player (i.e. playing strategies). Moreover, the OLL's ability to generate interest was tested over the game's complexity which corresponds to five, four and three *Dogs* environments for Dead End. Results obtained from these experiments demonstrate the approach's generality since interesting games emerge independently of game complexity, initial opponent behavior and *Cat* type. Finally, it appears that high interest is emerged through the player's interaction with cooperative opponents, since teamwork features of the OLT behaviors are maintained during OLL. For all the above-mentioned reasons, we believe that such a mechanism will be able to produce interesting interactive opponents (i.e. games) against even the most complex human playing strategy.

For subsequent steps, the methods used here need to be tested on other dissimilar predator/prey games in order to provide more evidence for their generality, and the interest measure proposed needs to be cross-validated against human players. The two following chapters cover these issues respectively.

Chapter 6

Pac-Man

> "*I felt it would be too stressful for a human being like Pac-Man to be continually surrounded and hunted down. So I created the monsters' invasions to come in waves. They would attack and then they would retreat. As time went by they would regroup, attack, and disperse again. It seemed more natural than having constant attack.*"
>
> Toru Iwatani, creator of Pac-Man.

In the preceding chapter we saw the successful application of the OLL approach in generating interesting Dead End games. Additional experiments in a dissimilar predator/prey game would display the effectiveness of the proposed methodology over different games of the same genre and expand the applicability of the method. In this chapter, by using one of the most representative test-beds of this computer game genre, that is Pac-Man, and by focusing on the non-player characters' behavior, we display the on-line learning mechanism's ability to increase the game's interest as well as maintain that interest at high levels while the game is being played. OLL demonstrates high robustness and adaptability to changing hand-crafted player strategies in a set of playing stages differing in complexity and topology.

More specifically, we test the proposed on-line learning mechanism's ability to generate interesting games over a number of computer game axes. These axes include (a) the (high-level) concept of the game; (b) the environment of the game and particularly the complexity of the game world (i.e. stage) and its topological features; (c) the opponents' behavior when the game starts and (d) the player's gaming skills. Experiments presented here demonstrate the generality of the methodology over the aforementioned

axes and verify our hypothesis that the proposed on-line learning mechanism constitutes a generic tool for obtaining predator/prey games of high entertainment independently of game type, game complexity and topology, initial opponent behavior and player.

The structure of the chapter is as follows. The Pac-Man test-bed used is described in Section 6.1 and the challenges for opponent behaviors of high performance in this game are mentioned in Section 6.2. The quantification process of the interest metric for this game is presented in Section 6.3. In addition, a way to measure performance is presented in Section 6.4. The off-line and on-line learning mechanisms are presented in the following two sections (Section 6.5 and Section 6.6) while results obtained from off-line and on-line learning experiments are presented in Section 6.7 and Section 6.8 respectively. Section 6.9 presents the experiment for further testing the on-line learning mechanism's ability to adapt against unknown playing strategies in this game. Finally, Section 6.10 outlines the Pac-Man experiments and Section 6.11 concludes the chapter through a comparison of the results obtained from both Pac-Man and Dead End.

6.1 The Pac-Man Game

The computer game test-bed studied in this chapter is a modified version of the original Pac-Man computer game released by Namco. The player's (*PacMan*'s) goal is to eat all the pellets appearing in a maze-shaped stage while avoiding being killed by four *Ghosts*. The game is over when either all pellets in the stage are eaten by *PacMan* or *Ghosts* manage to kill *PacMan*. In that case, the game restarts from the same initial positions for all five characters.

Compared to commercial versions of the game a number of features (e.g. power-pills) are omitted for simplicity; these features do not qualitatively alter the nature of 'interesting' in games of low interest. Cross-validation of this statement appears through the judgement and the beliefs of human players of both the original and this version of the game (see Chapter 7).

As stressed before, the Pac-Man game is investigated from the viewpoint of *Ghosts* and more specifically how *Ghosts'* emergent behaviors can contribute to the interest of the game.

Pac-Man — as a computer game domain for emerging interesting behaviors — is a

two-dimensional, multi-agent, grid-motion, predator/prey game. The game field (i.e. stage) consists of corridors and walls. Both the stage's dimensions and its maze structure are predefined. For the experiments presented here we use a 19×29 grid maze-stage where corridors are 1 grid-cell wide (see Figure 6.1(a) for an example stage).

The characters visualized in the Pac-Man game (as illustrated in Figure 6.1(a)) are a white circle that represents *PacMan* and 4 ghost-like characters representing the *Ghosts*. Additionally, there are black squares that represent the pellets and dark grey blocks of walls.

PacMan moves at double the *Ghosts'* speed and since there are no dead ends, it is impossible for a single *Ghost* to complete the task of killing it. Since *PacMan* moves faster than a *Ghost*, the only effective way to kill *PacMan* is for a group of *Ghosts* to hunt cooperatively. It is worth mentioning that one of *Ghosts'* properties is permeability. In other words, two or more *Ghosts* can simultaneously occupy the same cell of the game grid.

The simulation procedure of the Pac-Man game is as follows. *PacMan* and *Ghosts* are initially placed in the game field so that there is a suitably large distance between them. Then, the following occur at each simulation step:

1. Both *PacMan* and *Ghosts* gather information from their environment.
2. *PacMan* and *Ghosts* take a movement decision every simulation step and every second simulation step respectively; that is how *PacMan* achieves double the *Ghost*'s speed.
3. If the game is over (i.e. all pellets are eaten, *PacMan* is killed, or the simulation step is greater than a predetermined large number), then a new game starts from the same initial positions.
4. Statistical data such as number of pellets eaten, simulation steps to kill *PacMan* as well as the total *Ghosts'* visits to each cell of the game grid are recorded.

6.1.1 Stages

Similarly to the Dead End game (see Chapter 5), in this chapter we will attempt to test the on-line learning mechanism's ability to generate interesting Pac-Man games over

stages of different complexity and, furthermore, over stages of dissimilar topology[1].

6.1.1.1 Complexity

In order to distinguish between stages of different complexity, we require an appropriate measure to quantify this feature of the stage. This measure is

$$C = \frac{1}{E\{L\}} \tag{6.1}$$

where C is the complexity measure and $E\{L\}$ is the average corridor length of the stage. A "corridor" is defined by a path between two junctions on the stage.

According to (6.1), complexity is inversely proportional to the average corridor length of the stage. That is, the longer the average corridor length, the easier for the *Ghosts* to block *PacMan* and, therefore, the less complex the stage.

Figure 6.1 illustrates the four different stages used for the experiments presented here. Complexity measure values for the Easy A, Easy B, Normal and Hard stages are 0.16, 0.16, 0.22 and 0.98 respectively. Furthermore, given the Pac-Man game's conceptual features:

- blocks of walls should be included,
- corridors should be 1 grid-square wide,
- dead ends should be absent,

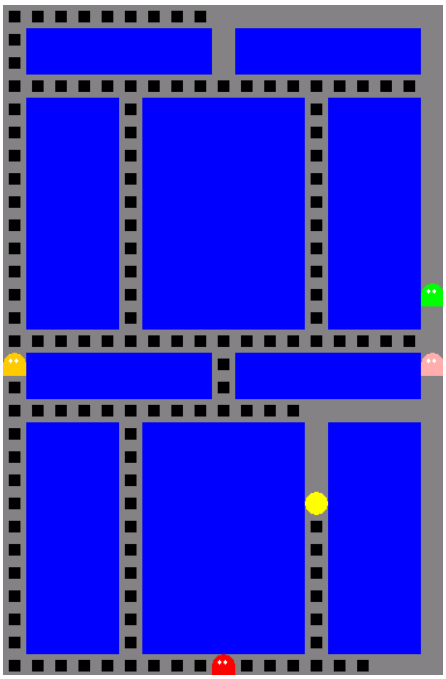
and the chosen size constraints of 19×29 cells, Hard stage is the most complex Pac-Man stage for the *Ghosts* to play.

6.1.1.2 Topology

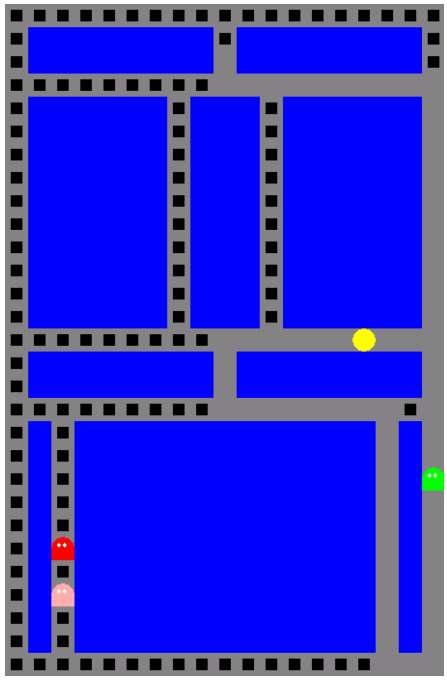
Stages of the same complexity, measured by (6.1), can differ in topology (i.e. layout of blocks on the stage). Thus, in the case of Easy A and Easy B (see Figure 6.1), stages have the same complexity value but are topologically different.

Overall, the choice of these four stages is made so as to examine the on-line learning approach's ability to emerge interesting opponents in stages of different complexity or

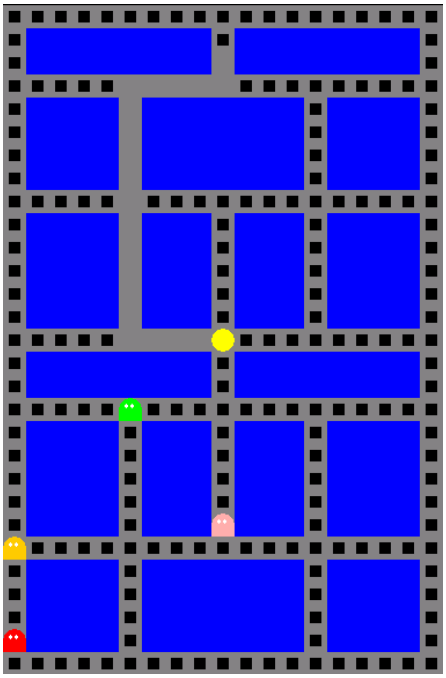
[1] This material was previously published in (Yannakakis and Hallam, 2004a; 2005b; 2005d)



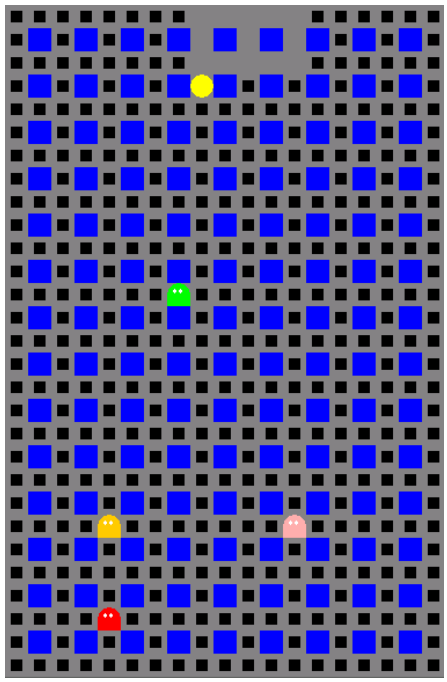
(a) Easy A



(b) Easy B



(c) Normal



(d) Hard

Figure 6.1: The four different stages of the Pac-Man game.

equally complex stages of different topology. Results presented in Section 6.8 show that the mechanism's effectiveness is independent of both the stage complexity and stage topology and, furthermore, illustrate the approach's generality for the game.

6.1.2 PacMan

Both the difficulty and, to a lesser degree, the interest of the game are directly affected by the intelligence of the *PacMan* player. We chose three fixed *Ghost*-avoidance and pellet-eating strategies for the *PacMan* player, differing in complexity and effectiveness. Each strategy is based on decision making applying a cost or probability approximation to the player's 4 neighbor cells (i.e. up, down, left and right). Even though the initial positions are constant, the non-deterministic motion of *PacMan* provides lots of diversity within games.

- Cost-Based (CB) *PacMan*: The CB *PacMan* moves towards its neighbor cell of minimal cost. Cell costs are assigned as follows: a cell with a pellet (pellet cell) costs 0; an empty cell costs 10; a cell occupied by a *Ghost* (*Ghost* cell) costs 100; a *Ghost*'s 4 neighbor cells cost 50 each. Wall cells are not assigned any cost and are ignored by *PacMan*. In case of equal minimal neighbor cell costs (e.g. two neighbor cells with pellets), the CB *PacMan* makes a random decision with equal probabilities among these cells. In other words, the CB *PacMan* moves towards a cost minimization path that produces effective *Ghost*-avoidance and (to a lesser degree) pellet-eating behaviors but only in the local neighborhood.

- Rule-Based (RB) *PacMan*: The RB *PacMan* is a CB *PacMan* plus an additional rule for more effective and global pellet-eating behavior. This rule can be described as follows. If all *PacMan*'s neighbor cells are empty (cost 10), then the probability of moving towards each one of the available directions (i.e. not towards wall cells) is inversely proportional to the distance (measured in grid-cells) to the closest pellet on that direction.

- Advanced (ADV) *PacMan*: The ADV *PacMan* checks in every non-occluded direction for *Ghosts*. If there is at least one *Ghost* in sight, then the probability of moving towards each one of the available directions is directly proportional to the distance to a *Ghost* in that direction. If there is no *Ghost* in sight, then the ADV *PacMan* behaves like RB *PacMan*. The ADV moving strategy is expected

to produce better global *Ghost*-avoidance behavior built upon the RB *PacMan*'s good pellet-eating strategy.

6.1.3 Neural Controlled Ghosts

The arcade version of Pac-Man uses a handful of very simple rules and scripted sequences of actions combined with some random decision-making to make the *Ghosts'* behavior less predictable. The game's interest decreases at the point where *Ghosts* are too fast to beat (Rabin, 2002). In our Pac-Man version, we require *Ghosts* to keep learning and constantly adapting to the player's strategy instead of being opponents with fixed strategies.

A feedforward neural controller is employed to manage the *Ghosts'* motion and is described in this section. Apart from the neural controller, three fixed (non-evolving) ways of controlling the *Ghosts* are presented in Section 6.1.4.

6.1.3.1 Input

Using their sensors, *Ghosts* inspect the environment from their own point of view and decide their next action. Each *Ghost* receives input information from its environment expressed in the neural network's input array of dimension 4 (see Figure 6.2). The input array consists of the relative coordinates from (a) *PacMan* in x ($\Delta_{x,P} = x_g - x_p$) and y ($\Delta_{y,P} = y_g - y_p$) axis and (b) the closest *Ghost* in x ($\Delta_{x,C} = x_g - x_c$) and y ($\Delta_{y,C} = y_g - y_c$) axis; where (x_g, y_g), (x_p, y_p) and (x_c, y_c) are the cartesian coordinates of the current *Ghost*'s, *PacMan*'s and closest *Ghost*'s current position respectively. *Ghost*'s input includes information for only one neighbor *Ghost* as this constitutes the minimal information for emerging cooperative behaviors. As in the Dead End game, we deliberately exclude from consideration any global sensing, e.g. information about the dispersion of the *Ghosts* as a whole, because we are interested specifically in the minimal sensing scenario.

All input values are linearly normalized into [0, 1] via $0.5[(\Delta_{i,J}/L_i) + 1]$ where $i \in \{x, y\}$, $J \in \{P, C\}$ and L_x, L_y are the width and height of the stage respectively.

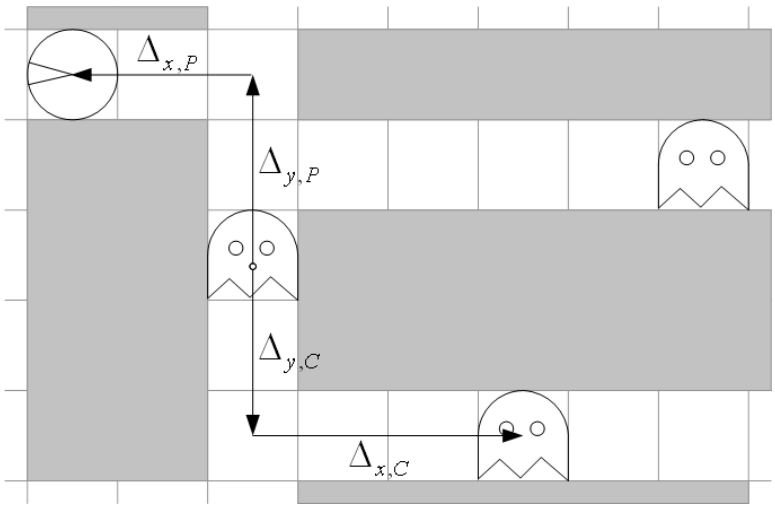


Figure 6.2: *Ghost's* environment perception.

6.1.3.2 Architecture

As stated before, a multi-layered fully connected feedforward neural network has been used for the experiments presented here (as shown in Figure 6.3). The sigmoid function is employed at each neuron.

The connection weights take values from -5 to 5 while the neural network's output is a four-dimensional vector (o_u, o_d, o_l, o_r) with respective values from 0 to 1 that represents the *Ghost*'s four movement options (up, down, left and right respectively). Each *Ghost* moves towards the available — unobstructed by walls — direction represented by the highest output value. Available movements include the *Ghost*'s previous cell position.

6.1.4 Fixed Strategy Ghosts

Apart from the neural controlled *Ghosts*, three additional non-evolving strategies have been tested for controlling the *Ghost*'s motion. These strategies are used as baseline behaviors for comparison with any neural controller emerged behavior.

- Random (R): *Ghosts* that randomly decide their next available movement. Available movements have equal probabilities of being picked.
- Followers (F): *Ghosts* designed to follow *PacMan* constantly. Their strategy is based on moving so as to reduce the greatest of their relative coordinates

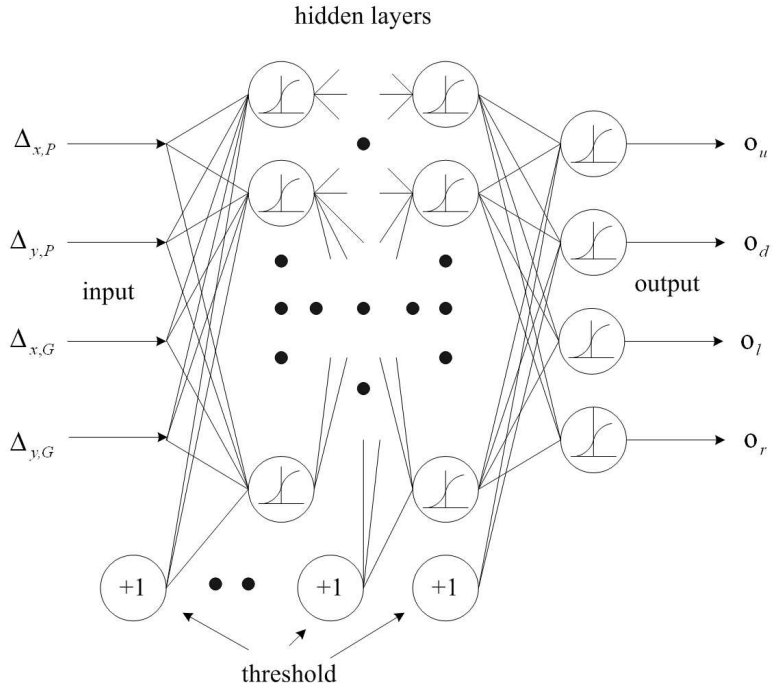


Figure 6.3: Multi-layer feedforward neural network controller of the *Ghosts*.

$(\Delta_{x,P}, \Delta_{y,P})$ from *PacMan*.

- Near-Optimal (O): A *Ghost* strategy designed to produce attractive forces between *Ghosts* and *PacMan* as well as repulsive forces among the *Ghosts*. For each *Ghost* X and Y values are calculated as follows.

$$\begin{aligned} X &= \text{sign}[\Delta_{x,P}]h(\Delta_{x,P}, L_x, 0.25) \\ &- \text{sign}[\Delta_{x,C}]h(\Delta_{x,C} - 1, L_x, 10) \qquad (6.2) \\ Y &= \text{sign}[\Delta_{y,P}]h(\Delta_{y,P}, L_y, 0.25) \\ &- \text{sign}[\Delta_{y,C}]h(\Delta_{y,C} - 1, L_y, 10) \qquad (6.3) \end{aligned}$$

where sign[τ]=$\tau/|\tau|$ and $h(\tau, \tau_m, p) = [1 - (|\tau|/\tau_m)]^p$. X and Y values represent the axis on which the near-optimal *Ghost* will move. Hence, the axis is picked from the maximum of $|X|$ and $|Y|$ while, the direction is decided from this value's sign. That is, if $|X| > |Y|$, then go right if sign$[X] > 0$ or go left if sign$[X] < 0$; if $|Y| > |X|$, then go up if sign$[Y] > 0$ or go down if sign$[Y] < 0$.

6.2 Challenges

The challenges for a *Ghost* to achieve behaviors of high performance in Pac-Man are consistent with the respective challenges of a *Dog* in Dead End (see Section 5.2) and a Human in *FlatLand* (see Section 4.6). These challenges are briefly as follows:

- Pac-Man is a fully dynamic multi-agent game environment.
- *Ghosts*' communication is partial and implicit while their perceived information is discontinuous time-varying.
- Cooperative action must be built on this kind of communication.

6.3 Interest Parameter Values

In this section we present the procedures followed to obtain the appropriate parameter values of the interest estimate (2.5) for the Pac-Man game.

6.3.1 Minimum Playing Time — t_{min}

For the experiments presented here, t_{min} is 32 for the Easy stage, 35 for the Normal stage and 63 for the Hard stage, which is obtained as the minimum simulation time that *PacMan* survives when playing against the best-performing Near-Optimal *Ghosts* in each stage.

6.3.2 Maximum Playing Time — t_{max}

As previously defined in Chapter 2, t_{max} is the maximum evaluation period of play, or else the maximum lifetime of the player. For Pac-Man this number corresponds to the minimum simulation period required by the RB *PacMan* (best pellet-eater) to clear the stage of pellets. In the experiments presented here t_{max} is 300 for the Easy stage. Given that the Easy stage contains 187 pellets, 113 steps (i.e. 37.6% of the playing period) correspond to backtrack movement decisions of the *PacMan*. The difference between the estimated (i.e. evaluation period) and the real (i.e. pellets of the stage plus number of backtrack moves) evaluation period is illustrated in Figure D.1 in Appendix D when

a RB *PacMan* plays in a stage without *Ghosts*. The minimum number of pellets left by the RB *PacMan* is also displayed in the same figure.

Thus, given the evaluation period's percentage of backtrack movements in the Easy stage ($B_E = 37.6\%$), the backtracking period (B_i) for each stage is computed via (6.4)

$$\frac{E\{L_1\}}{E\{L_2\}} = \frac{B_1}{B_2} \tag{6.4}$$

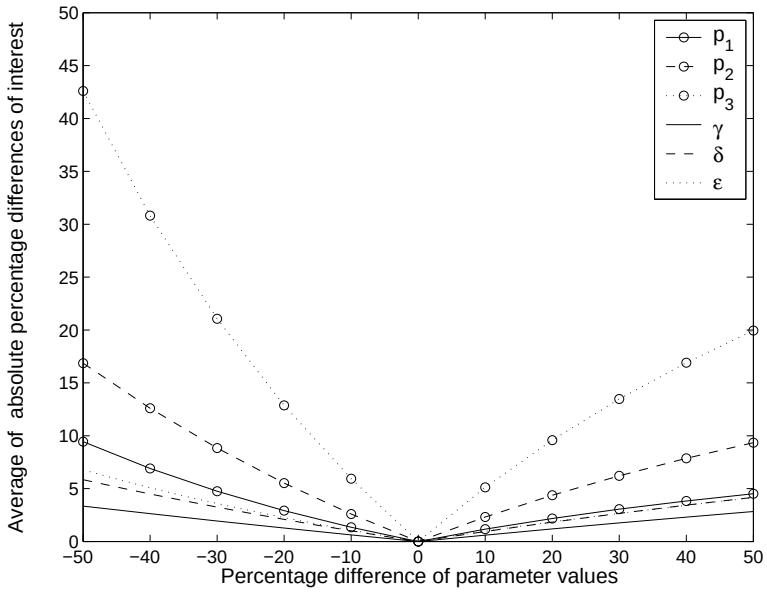
since it should be linearly proportional to the average corridor length of the stage. Consequently, we come up with t_{max} values of 320 for the Normal stage (227 pellets on the stage) and 466 for the Hard stage (425 pellets).

6.3.3 Weighting Parameters

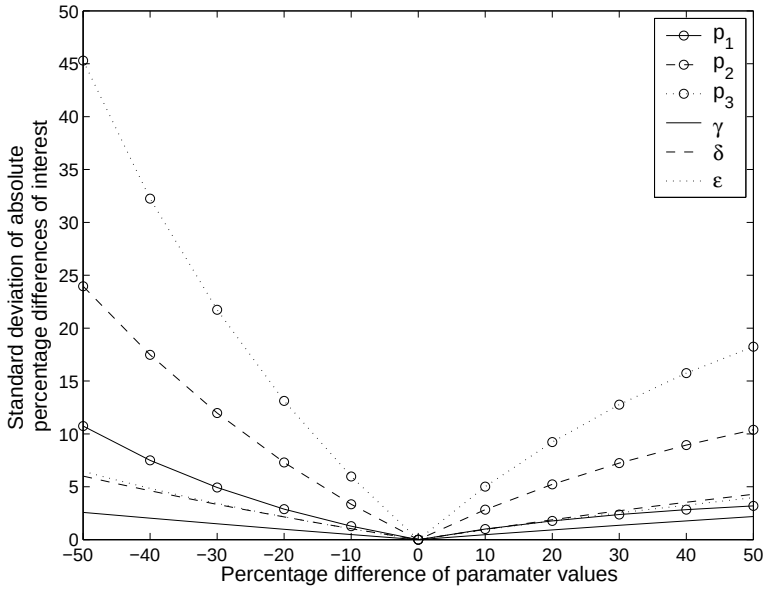
In order to obtain values for the interest formula weighting parameters γ, δ and ε we select empirical values based on the specific game. In Pac-Man, spatial diversion of the opponents is of the greatest interest. The game no longer engages the player when *Ghosts* stick in a corner instead of wandering around the stage. Thus, diversity in gameplay (S) and challenge (T) should come next in the importance list of interest criteria. Given the above-mentioned statements and by adjusting these three parameters so that the interest value escalates as the opponent behavior changes from Random to Near-Optimal and then to Follower, we come up with $\gamma = 1, \delta = 2$ and $\varepsilon = 3$.

By following the sensitivity analysis procedure described in Section 5.3.3 for the Dead End game, we obtain the function between the absolute percentage differences of the interest value and the percentage differences of the interest weighting parameters for the Pac-Man game (see Figure 6.4).

Similarly to the Dead End game, changes on the p_1, p_2 and p_3 parameters seem to affect the I value more than γ, δ and ε. More specifically, p_2 and p_3 reveal significant differences (i.e. greater than 5%) in I when decreased by 15% (i.e. $p_2 = 0.85$) and 9% (i.e. $p_3 = 3.64$) or increased by 20% (i.e. $p_2 = 1.2$) and 10% (i.e. $p_3 = 4.4$) respectively. For p_1 significant change in I is observed only when decreased by up to 35% (i.e. $p_1 = 0.325$). Accordingly, both ε and δ parameters reveal significant differences in I only when decreased by 40% and 45% respectively. Finally, for γ no significant change in I is observed even when changed by up to 50%.



(a) Average



(b) Standard deviation

Figure 6.4: Average and standard deviation of absolute percentage differences of I over ten runs for each weighting parameter.

As in the Dead End game and as far as mainly p_2 and p_3 are concerned, their selected values project a rather robust I value considering that they constitute power parameters in (2.5).

6.4 Performance Measurement

When a predator/prey game is investigated from the predator's viewpoint, optimality can be measured in the predators' ability to kill the prey. Thus, a predator's behavior that always manages to kill the prey in such games is obviously a desired behavior in terms of optimality.

Prey-killing ability is the primary factor that determines how good a behavior is (i.e. its performance) in the Pac-Man game as well. Furthermore, the behavior of preventing *PacMan* from eating pellets constitutes an additional factor of the desired optimal behavior. This behavior also implies a fast-killing behavior, which is also desired from optimal predators. Given these, a measure designed to give an approximation of a group of *Ghosts'* performance over a specific number N_p of games played, is

$$P = \frac{\zeta \frac{\mathrm{K}}{N_p} + \eta \min\left\{1 + \left(\frac{e_{min} - E\{e\}}{e_{max} - e_{min}}\right), 1\right\}}{\zeta + \eta} \tag{6.5}$$

where P is the performance of a *Ghost* group behavior taking values from 0 to 1; K is the number of *PacMan* kills within N_p games; $E\{e\}$ is the average number of pellets eaten by *PacMan* over the N_p games; ζ, η are weight parameters (for the experiments in the Pac-Man game $\zeta = \eta = 1$); e_{min}, e_{max} are the lower and upper bound estimates of the eaten pellets e respectively.

The lower bound of the eaten pellets corresponds to the minimum number of pellets eaten by any *PacMan* type when playing against the Near-Optimal *Ghosts*. Thus $e_{min} = 70$ for the Easy stage; $e_{min} = 80$ for the Normal stage and $e_{min} = 100$ for the Hard stage. Likewise, e_{max} corresponds to the maximum number of pellets that may be eaten, that is the number of pellets that each stage contains by design — $e_{max} = 187$, $e_{max} = 227$ and $e_{max} = 425$ for the Easy, Normal and Hard stages respectively.

6.5 Off-Line Learning

We use an off-line evolutionary learning approach in order to produce some 'good' (i.e. well-performing) initial behaviors. An additional aim of the proposed algorithm is to generate dissimilar behaviors of high fitness — varying from blocking to aggressive (see Section 6.7) — offering diverse seeds for the on-line learning mechanism in its attempt to generate emergent *Ghost* behaviors that make the game interesting.

According to the mechanism, each *Ghost* has a genome that encodes the connection weights of its neural network. The evolving process is limited to the connection weights. A population of 80 neural networks (*Ghosts*) is initially generated with uniformly distributed random connection weights that lie within [-5, 5]. Then, based on the GGA approach (see Section 4.1.1.1) — with specific adjustments for the Pac-Man game only in the GGA's first two basic steps (evaluation steps) — the off-line learning algorithm is as follows:

Step 1: Every *Ghost* in the population is cloned 4 times. These 4 clones are placed in the Pac-Man game field and play N_t games (in the experiments presented here $N_t = 10$ games), each one for an evaluation period of e_p simulation steps. The outcome of these games is to ascertain the time taken to kill *PacMan* t_k for each game.

Step 2: Each *Ghost* is evaluated via (6.6) for each game and its fitness value is given by $E\{f\}$ over the N_t games.

$$f = [1 - (t_k/e_p)]^{\frac{1}{4}} \tag{6.6}$$

By the use of the f fitness function, that takes values from 0 to 1, we promote *PacMan*-killing behaviors capable of achieving high performance values P.

The algorithm is terminated when a predetermined number of generations $\mathbf{T}$ is achieved (e.g. $\mathbf{T} = 1000$) and the fittest *Ghost*'s connection weights are saved. *Ghosts* play few games (i.e. $N_t = 10$) when evaluated by the off-line learning method. Even though this evaluation procedure constitutes an approximation of the examined *Ghost*'s overall performance in a greater number of games, it keeps the computational cost low.

6.6 On-Line Learning

As previously noted, games which can learn and adapt to new playing strategies offer a richer interaction to entertain the player. For that purpose we use an evolutionary machine learning mechanism for the Pac-Man game which is based on the idea of *Ghosts* that learn while they are playing against *PacMan*. Or else, *Ghosts* that are reactive to any player's behavior and learn from its strategy instead of being the predictable and somewhat uninteresting characters that exist in all versions of this game today. Furthermore, this approach's additional objectives are to keep the game's interest at high

levels as long as it is being played and to be able to achieve good real-time performance (i.e. low computational effort during gameplay). This approach, which is built upon the algorithm presented in Section 4.3, is first applied in Chapter 5 for the Dead-End game and its modified version for the Pac-Man game is presented here.

Beginning from any initial group of OLT *Ghosts*, the OLL mechanism transforms them into a group of heterogeneous opponents that are conceptually more interesting to play against. An OLT *Ghost* is cloned four times and its clones are placed in the game field to play against a selected fixed *PacMan* type in a selected stage. Then, the OLL approach follows the basic steps of the algorithm presented in Section 4.3 with adjustments for the Pac-Man game:

Step 1: Each *Ghost* is evaluated every e_p simulation steps via (6.7), while the game is played — e_p is 25 simulation steps.

$$f' = \sum_{i=1}^{e_p} \left\{ d_{P,2i} - d_{P,(2i-1)} \right\} \tag{6.7}$$

where $d_{P,i}$ is the distance between the *Ghost* and *PacMan* at the i simulation step. This fitness function promotes *Ghosts* that move towards *PacMan* within an evaluation period of e_p simulation steps. In other words, this function represents the intention to kill *PacMan*.

Step 2: A pure elitism selection method is used where only the fittest solution is able to breed. The fittest parent clones an offspring with a probability p_c that is inversely proportional to the normalized cell visit entropy (i.e. $p_c = 1 - H_n$) given by (2.4). In other words, the higher the cell visit entropy of the *Ghosts*, the lower the probability of breeding new solutions. If there is no cloning, then go back to Step 1, else continue to Step 3.

Step 3: Mutation occurs in each gene (connection weight) of the offspring's genome with a small probability p_m (e.g. 0.02). A gaussian random distribution is used to define the mutated value of the connection weight. The mutated value is obtained from (6.8).

$$w_m = \mathcal{N}(w, 1 - H_n) \tag{6.8}$$

where w_m is the mutated connection weight value and w is the connection weight value to be mutated. The gaussian mutation, presented in (6.8), suggests that the higher the normalized entropy of a group of *Ghosts*, the smaller the variance of

the gaussian distribution and therefore, the less disruptive the mutation process as well as the finer the precision of the GA.

Step 4: The mutated offspring is evaluated briefly via (6.7) in off-line mode, that is, by replacing the least-fit member of the population and playing an off-line (i.e. no visualization of the actions) short game of e_p simulation steps. If there is a human playing Pac-Man, then the *PacMan*'s motion trail of the last e_p simulation steps is recorded and opponents are evaluated against it in off-line mode. The fitness values of the mutated offspring and the least-fit *Ghost* are compared and the better one is kept for the next generation. This pre-evaluation procedure for the mutated offspring attempts to minimize the probability of group behavior disruption by low-performance mutants. The fact that each mutant's behavior is not tested in a single-agent environment but within a group of heterogeneous *Ghosts* helps more towards this direction. If the least-fit *Ghost* is replaced, then the mutated offspring takes its position in the game field as well (i.e. replacement method).

The algorithm is terminated when a predetermined number of games has been played or a game of high interest (e.g. $I \geq 0.7$) is found.

We mainly use short simulation periods ($e_p = 25$) in order to evaluate *Ghosts* in OLL aiming to the acceleration of the on-line evolutionary process. The same period is used for the evaluation of mutated offspring; this is based on two primary objectives: 1) to apply a fair comparison between the mutated offspring and the least-fit *Ghost* (i.e. same evaluation period) and 2) to avoid undesired high computational effort in on-line mode (i.e. while playing).

6.7 Off-Line Learning Experiments

The experiment presented in this section is focused on producing well-behaved *Ghosts* in terms of the performance measure described in Section 6.4. For each of the four stages examined, we train *Ghosts* against all three types of *PacMan* player through the off-line learning mechanism presented in Section 6.5. As in the Dead End game, the neuro-controller used here is the 5-hidden neuron ANN. For obtaining minimal *Ghost* controllers capable of achieving high performances, forty trials of the modified 1-hidden layer ECWAS (see Section 4.8.2.1) were held in the Normal stage against the

	Trained off-line by playing against					
	CB		RB		ADV	
	P	I	P	I	P	I
R	0.456	0.586	0.420	0.505	0.381	0.520
F	0.809	0.784	0.734	0.775	0.617	0.775
O	0.977	0.683	0.917	0.719	0.989	0.678
B	0.984	0.458	0.743	0.554	0.904	0.476
A	0.699	0.646	0.662	0.613	0.764	0.512
H	0.549	0.297	0.504	0.478	0.464	0.356
$E\{\}$	0.746	0.576	0.663	0.607	0.686	0.553

Table 6.1: Easy A stage: Performance (P) and Interest (I) values (average values of 10 samples of 50 games each) of fixed strategy (R, F, O) and OLT *Ghosts* (B, A, H) playing against all three *PacMan* types (CB, RB, ADV). Average P and I values ($E\{\}$) of all six strategies appear in the bottom row. Experiment Parameters: $N_p = 50$, population size is 80, $g = 1000$, $e_p = 300$ simulation steps, $N_t = 10$ games, $p_m = 0.02$, 5-hidden neurons controller.

	Trained off-line by playing against					
	CB		RB		ADV	
	P	I	P	I	P	I
R	0.491	0.583	0.384	0.560	0.357	0.460
F	0.519	0.707	0.506	0.695	0.441	0.682
O	0.938	0.649	0.904	0.591	1.000	0.525
B	0.893	0.498	0.897	0.559	0.990	0.502
A	0.787	0.624	0.694	0.561	0.653	0.552
H	0.504	0.324	0.521	0.441	0.417	0.381
$E\{\}$	0.689	0.564	0.625	0.567	0.622	0.512

Table 6.2: Easy B stage: Performance and Interest values. See the caption of Table 6.1 for the experiment parameters.

ADV *PacMan*. ECWAS automatically designed the 5-hidden neuron ANN more times than any other architecture (i.e. 8 out of 40 trials) with the highest average performance (0.682).

	Trained off-line by playing against					
	CB		RB		ADV	
	P	*I*	*P*	*I*	*P*	*I*
R	0.423	0.547	0.363	0.586	0.356	0.523
F	0.754	0.771	0.701	0.772	0.621	0.771
O	0.891	0.729	0.897	0.749	0.964	0.686
B	0.734	0.576	0.689	0.412	0.869	0.442
A	0.661	0.654	0.606	0.652	0.662	0.555
H	0.348	0.190	0.310	0.250	0.467	0.423
$E\{\}$	0.635	0.578	0.592	0.570	0.656	0.566

Table 6.3: Normal stage: Performance and Interest values. See the caption of Table 6.1 for the experiment parameters.

	Trained off-line by playing against					
	CB		RB		ADV	
	P	*I*	*P*	*I*	*P*	*I*
R	0.364	0.390	0.210	0.390	0.304	0.388
F	0.573	0.772	0.531	0.754	0.650	0.762
O	0.812	0.692	0.788	0.711	0.696	0.492
B	0.822	0.508	0.707	0.636	0.577	0.524
A	0.654	0.612	0.694	0.687	0.545	0.650
H	0.496	0.434	0.466	0.539	0.488	0.595
$E\{\}$	0.620	0.568	0.571	0.625	0.543	0.568

Table 6.4: Hard stage: Performance and Interest values. See the caption of Table 6.1 for the experiment parameters.

The off-line training experiment is described as follows.

- Apply the off-line learning mechanism playing against each type of *PacMan* player separately.
- *Ghosts* trained against a specific type of *PacMan* player are evaluated by playing 100 non-evolution games against the same *PacMan* type.
- Apply the bootstrapping procedure presented in Appendix A ($N = 50$) to deter-

mine the interest and performance values' confidence intervals.

In each of Table 6.1, Table 6.2, Table 6.3 and Table 6.4 the off-line learning experiment's outcome is displayed for Easy A, Easy B, Normal and Hard stages respectively. More comprehensively, in each of the aforementioned tables both the performance and the interest values of six different *Ghosts'* behaviors against all three different *PacMan* types as well as their average values $E\{\}$ are presented for comparison. In the first three rows of each table, the fixed strategy *Ghosts'* (i.e. R: Random, F: Followers, O: Near-Optimal) performance and interest values are presented against each type of *PacMan* player. Furthermore, in the three subsequent rows, P and I values of three different types of emergent behaviors playing against each *PacMan* type are displayed. More specifically, there are three different OLT *Ghosts* generated by playing against each *PacMan* type separately which produce nine different behaviors in total. These behaviors, which initially were distinguished empirically through visual inspection, are classified by their performance and interest values and described as follows.

- Blocking (B): These are the OLT *Ghosts* that achieve the best performance against each *PacMan* type. Their behavior is characterized as 'Blocking' because they tend to wait for *PacMan* to enter a specific area that is easy for them to block and then kill. Their average normalized cell visit entropy value $E\{H_n\}$ lies between 0.55 and 0.65.

- Aggressive (A): These are OLT *Ghosts* that achieve lower performance in comparison to the blockers. Their behavior is characterized as 'Aggressive' because they tend to follow *PacMan* all over the stage in order to kill it. This motion feature generates the highest I value ($E\{H_n\} \geq 0.65$) among the interest values generated by the three different emergent behaviors.

- Hybrid (H): These are suboptimal OLT *Ghosts* that achieve the lowest performance ($P \leq 0.55$) and low interest value in comparison to the aforementioned B and A *Ghosts* ($E\{H_n\} < 0.55$). Their behavior is characterized as 'Hybrid' because they tend to behave as a Blocking-Aggressive hybrid which proves to be ineffective at killing *PacMan*.

As far as the interest value generated by the above-mentioned behaviors is concerned, confidence intervals (± 0.0647 maximum, ± 0.0313 on average — see Table D.1 in Appendix D) obtained by the bootstrapping procedure presented in Appendix A indicate that B, A and H are significantly different.

According to the above-mentioned tables, Near-optimal and Blocking behavior *Ghosts* achieve high-performance values against all three *PacMan* types, whereas their interest value is not as high as their performance value. This illustrates the compromise between optimality and interest that has to be made because, in a predator/prey computer game, optimal killing behaviors are almost never interesting behaviors. On the other hand, Followers are likely to produce the most interesting behaviors (among the behaviors examined) for this game.

Viewing results from the *PacMan* type perspective (i.e. the average values in the bottom row of each table), no safe conclusion can be made for the effectiveness of the three *PacMan* types since it appears that *Ghosts*' performance against each type depends on the stage's complexity and/or topology. However, concerning the three *PacMan* types' generated interest, it seems that the RB type is the most interesting *PacMan* for the *Ghosts* to play against.

6.8 On-Line Learning Experiments

As previously mentioned, the off-line learning procedure is a mechanism that produces near-optimal solutions to the problem of killing *PacMan* and minimizing the pellets eaten in a game. These solutions are the OLL mechanisms' initial points in the search for more interesting games. In the following parts of this section comprehensive experiments on each of the four stages are presented.

6.8.1 Easy A Stage

The OLL experiment conducted in the Easy A stage is described as follows.

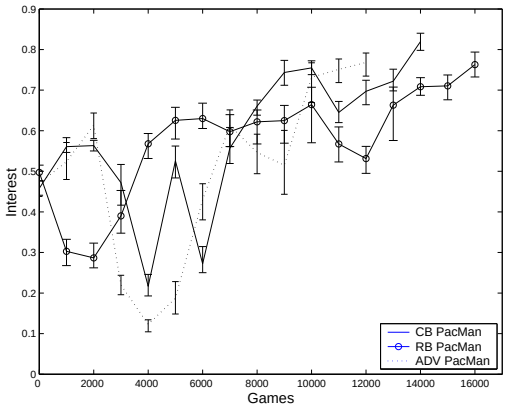
- Pick the nine different emerged *Ghosts'* behaviors produced from the off-line learning experiments presented in Section 6.7 (i.e. B, A and H behaviors emerged by playing against each *PacMan* type).
- Starting from each OLT behavior, apply the OLL mechanism by playing against each type of *PacMan* player separately. This makes a total of 27 different OLL attempts.

- Calculate the interest (bootstrapping procedure with $N = 50$ — see Appendix A) of the game every 100 games during each OLL attempt.

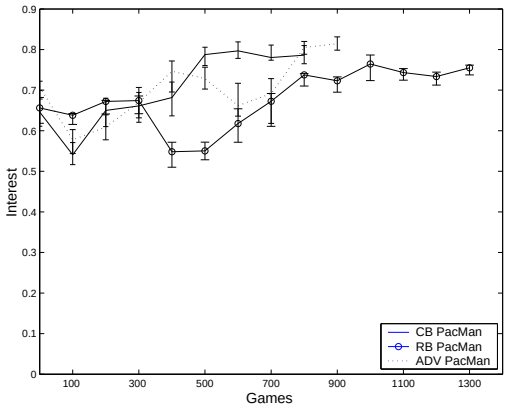
The outcome of this experiment is presented in Table 6.5 and Figure 6.5.

Figure 6.5 illustrates the overall picture of the OLL experiments for the Easy A stage. The evolution of interest over the OLL games of each one of the nine different OLT behaviors is presented in a sub-figure of Figure 6.5. For each sub-figure, three lines are illustrated, representing the interest values of the OLL attempt playing against the three different *PacMan* types (a total of 27 different OLL attempts).

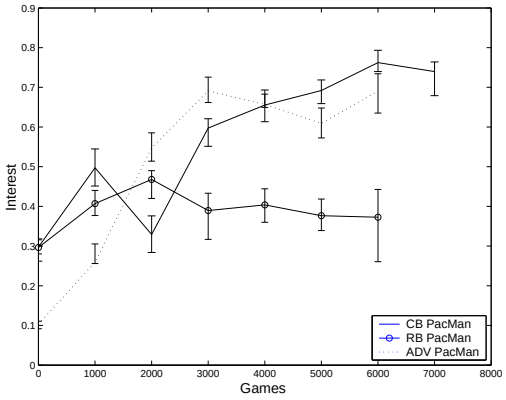
As seen from Figure 6.5, the OLL mechanism manages to find ways of increasing the interest of the game regardless of the initial OLT behavior or the *PacMan* player *Ghosts* play against. In all experiments presented here the learning mechanism is capable of producing games of higher than the initial interest as well as keeping that high interest for a long period. Such a fact demonstrates both the mechanism's robustness over a set of different initial behaviors and its adaptability against all *PacMan* types. There is obviously a slight probability of disruptive mutations (the higher the game's interest, through the cell visit entropy value, the less the probability of mutation) that can cause undesired drops in the game's interest. However, as seen from Figure 6.5, the learning mechanism is robust enough to recover from such disruptive phenomena. When the initial *Ghost* behavior is interesting (see Figures 6.5(b) and 6.5(e)) then the mechanism is likely to keep the game at these high, or ever higher, levels of interest. Given an interesting initial behavior (e.g. Aggressive behavior, $I > 0.6$) or even a suboptimal H behavior in some cases (see Figures 6.5(f)), it takes some hundreds of games (around 500 games in most cases) for the learning mechanism to produce games of high interest. On the other hand, it takes some thousand games to transform an uninteresting near-optimal blocking behavior (see Figure 6.5(a), Figure 6.5(d) and Figure 6.5(g)) into an interesting one. That is because the OLL process requires an initial long period to disrupt the features of an uninteresting blocking behavior in order to be able to increase the interest of the game. This long period of disruption appears when the initial on-line *Ghosts'* behavior (B, A, or H) is emerged by playing against ADV *PacMan* as well (see Figures 6.5(g), 6.5(h) and 6.5(i)). This appears to be likely because off-line training against ADV *PacMan* seems to produce the least interesting games (see Table 6.1).



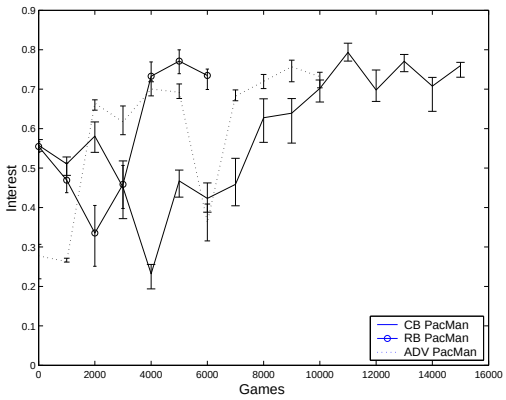
(a) B OLT against CB *PacMan*



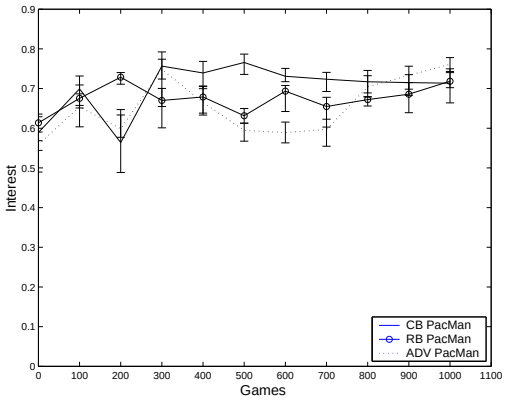
(b) A OLT against CB *PacMan*



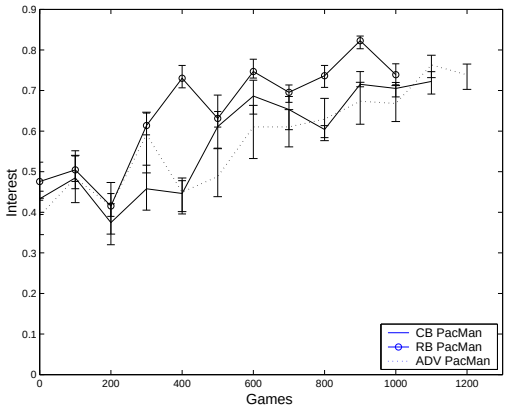
(c) H OLT against CB *PacMan*



(d) B OLT against RB *PacMan*

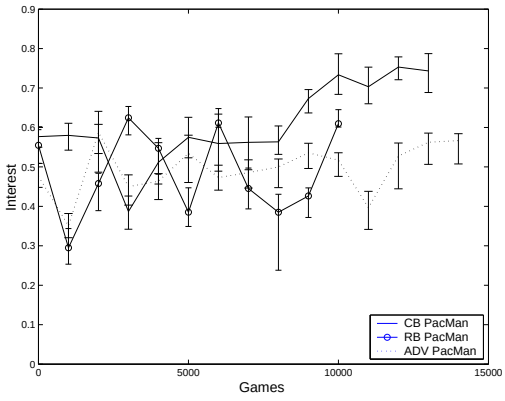


(e) A OLT against RB *PacMan*

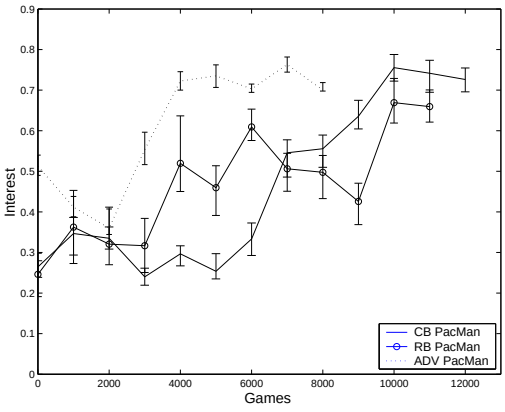


(f) H OLT against RB*PacMan*

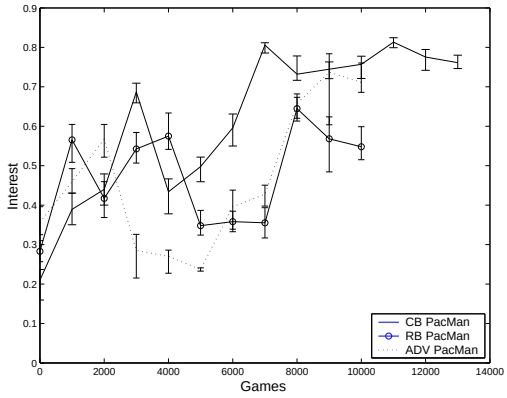
Table 6.5 presents the best average interest values obtained from the OLL mechanism. It is clear that the OLL approach constitutes a robust mechanism that, starting from



(g) B OLT against ADV *PacMan*



(h) A OLT against ADV *PacMan*



(i) H OLT against ADV *PacMan*

Figure 6.5: Easy A stage: Game interest (average and confidence interval values) over the number of OLL games. For reasons of computational effort, the OLL procedure continues for a number of games, large enough to illustrate the mechanism's behavior, after a game of high interest ($I \geq 0.7$) is found. Initial *Ghost* behaviors appear in sub-figure captions. Experiment Parameters: $e_p = 25$ simulation steps, $p_m = 0.02$, 5-hidden neurons controller.

near-optimal or suboptimal *Ghosts*, manages to emerge interesting games (i.e. interesting *Ghosts*) in the majority of cases (i.e. in 19 out of 27 cases $I > 0.7$). It is worth mentioning that in 15 out of 27 different OLL attempts the best interest value is significantly greater of statistically equal to the respective Follower's value (i.e. 0.784 against CB, 0.775 against RB and 0.775 against ADV).

			OLL - Playing against		
			CB	RB	ADV
Initial OLT behaviors	CB	B	0.8195	0.7605	0.7682
		A	0.7967	0.7644	0.8144
		H	0.7622	0.4678	0.6910
	RB	B	0.7933	0.7713	0.7570
		A	0.7657	0.7184	0.7609
		H	0.7224	0.8228	0.7634
	ADV	B	0.7532	0.6244	0.5667
		A	0.7554	0.6693	0.7630
		H	0.8133	0.6448	0.7374

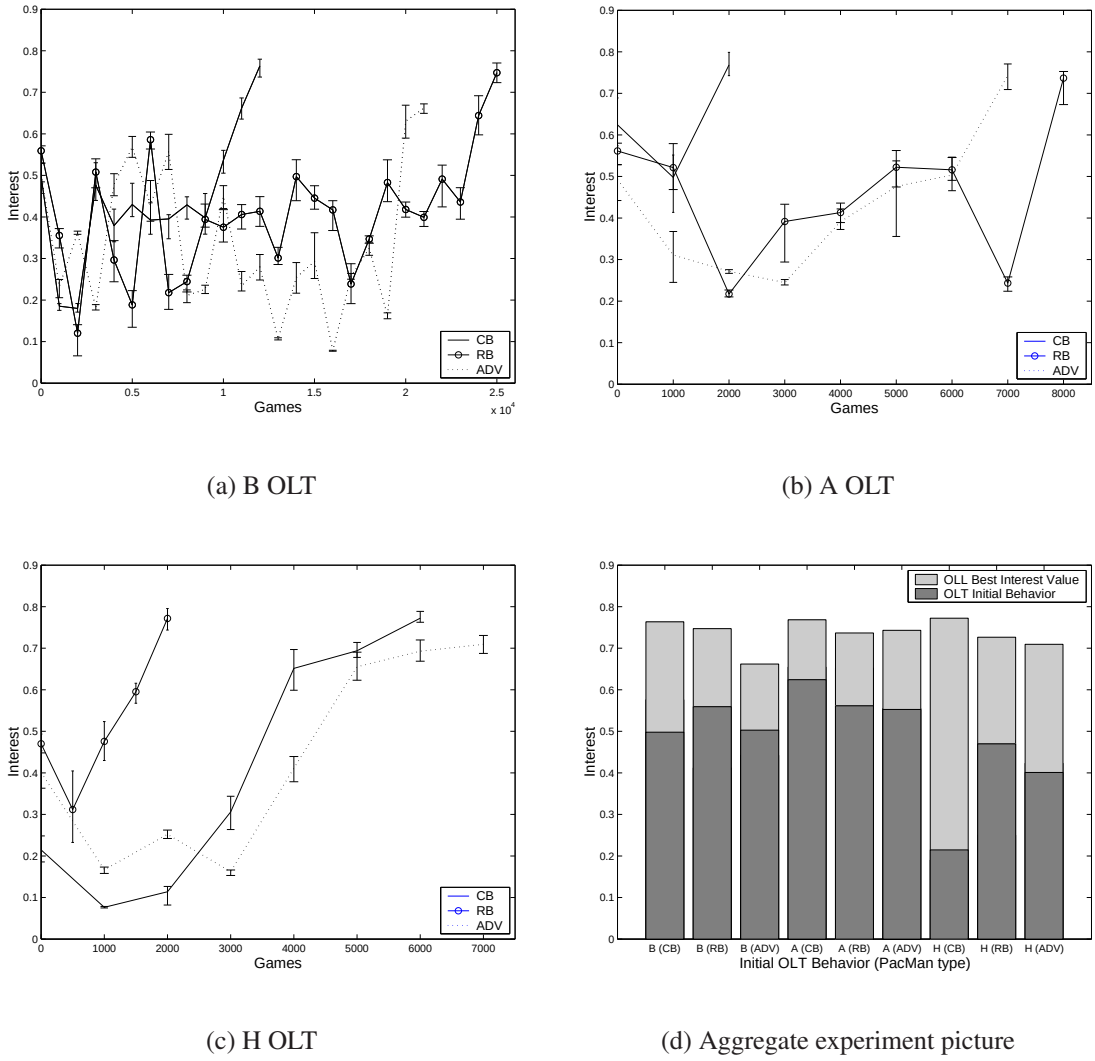
Table 6.5: Easy A stage: Best interest values achieved from on-line learning.

6.8.2 Easy B Stage

Even though Easy A and Easy B stages have the same complexity values, they are topologically dissimilar as already noted in Section 6.1.1.2. In principle, by design of the complexity measure, when two game environments are equally complex this corresponds to statistically equal average performance values obtained by the same opponents when playing in both stages (this is demonstrated experimentally in Table D.3 in Appendix D). Therefore, for the Easy B stage we choose to apply the OLL with initial OLT behaviors emerged from the Easy A stage. This experiment intends to demonstrate the effect of the topology of a stage in the interest of the game.

The OLL experiment follows the steps described for the Easy A stage in Section 6.8.1. However for this stage, starting from each OLT behavior, we apply the OLL mechanism by only playing against the same type of *PacMan* as was used off-line. This makes a total of 9, instead of 27, different OLL attempts. The complete set of 27 OLL attempts in the Easy A stage was primarily designed to demonstrate the mechanism's adaptability to new playing strategies. Since Easy B stage is devised to primarily demonstrate the effectiveness and robustness of the mechanism over stages of different topology, we believe that the set of OLL attempts for this stage is adequate to serve this purpose.

The evolution of interest over the OLL games of each one of the three different OLT



(a) B OLT

(b) A OLT

(c) H OLT

(d) Aggregate experiment picture

Figure 6.6: Easy B stage: Game interest over the number of OLL games. For reasons of computational effort, the OLL procedure continues for a number of games large enough to illustrate its behavior, after a game of high interest ($I \geq 0.7$) is found. Initial *Ghost* behaviors appear in (a), (b) and (c) sub-figure captions whereas (d) illustrates the overall picture of the experiment. Experiment Parameters: $e_p = 25$ simulation steps, $p_m = 0.02$, 5-hidden neurons controller.

behaviors is presented in a sub-figure of Figure 6.6. For each of the three sub-figures, three lines are illustrated, representing the interest values and their respective confidence intervals of the OLL attempt playing against the three different *PacMan* types. Figure 6.6(d) illustrates the overall picture of the OLL experiments by comparing the initial interest of the game against the best average interest value achieved from OLL.

It is apparent that the OLL approach constitutes a robust mechanism that, starting from suboptimal *Ghosts*, manages to emerge interesting games in the vast majority of cases (i.e. in 8 out of 9 cases $I > 0.7$). It is worth mentioning that in 8 out of 9 different OLL attempts the best interest value is significantly greater than or statistically equal to the respective Follower's value (i.e. 0.707 against CB, 0.695 against RB and 0.682 against ADV).

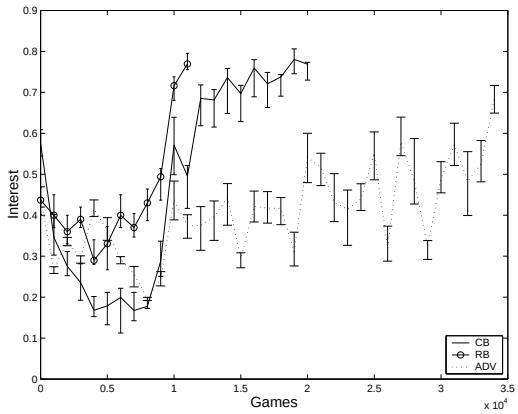
On-line learning enhances the interest of the game independently of the initial OLT behavior or the *PacMan* player they play against. In all experiments presented here the learning mechanism is capable of producing games of higher than the initial interest as well as keeping that high interest for a long period. As in Easy A stage experiments, there appears a probability of disruptive mutations that can cause undesired drops in the game's interest. However, as seen from Figure 6.6, OLL is robust enough to recover from such disruptive phenomena.

Finally, given an Aggressive or a Hybrid initial behavior it takes some few thousands of games for the learning mechanism to produce games of high interest. On the other hand, it takes some several thousand games to transform an uninteresting near-optimal blocking or behavior (see Figure 6.6(a)) into an interesting one. This follows our observations in the Easy A stage, where an uninteresting blocking (and hybrid in some cases) behavior required a long period to be transformed into an interesting one. However, differences in the time required by the OLL mechanism to produce highly interesting games can be found between the two Easy stages. In comparison to the Easy A stage, there is a noticeable delay of the mechanism when starting from an A OLT and playing against the CB and RB *PacMan* types in the Easy B stage. That could be explained through the lower initial interest values that these types of player generate in the particular stage.

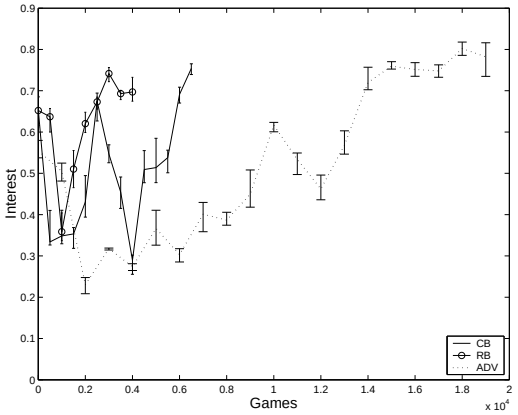
6.8.3 Normal Stage

Through the Normal stage we will explore the mechanism's effectiveness with respect to the complexity of the stage. We therefore follow the OLL experimental steps presented in Section 6.8.2 for the Easy B stage.

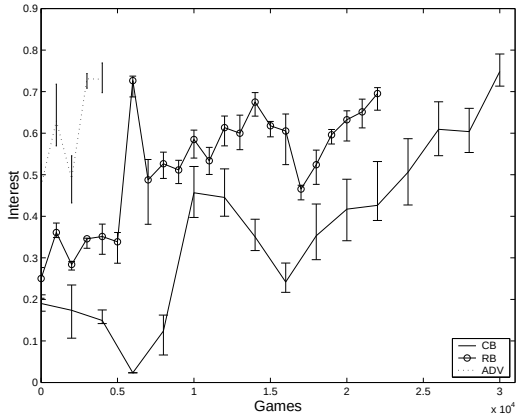
The experiment's outcome for the Normal Stage is illustrated in Figure 6.7 where the evolution of interest over the OLL games of each one of the three different OLT behav-



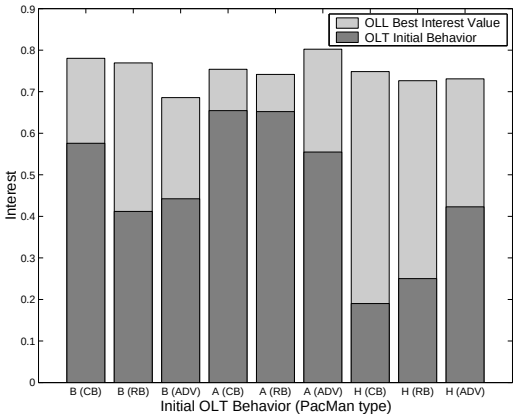
(a) B OLT



(b) A OLT



(c) H OLT



(d) Aggregate experiment picture

Figure 6.7: Normal stage: Game interest over the number of OLL games. For reasons of computational effort, the OLL procedure continues for a number of games large enough to illustrate its behavior, after a game of high interest ($I \geq 0.7$) is found. Initial *Ghost* behaviors appear in (a), (b) and (c) sub-figure captions whereas (d) illustrates the overall picture of the experiment. Experiment Parameters: $e_p = 25$ simulation steps, $p_m = 0.02$, 5-hidden neurons controller.

iors is presented in each of its sub-figures. Figure 6.7(d) illustrates the overall picture of the OLL experiments by comparing the initial interest of the game against the best average interest value achieved from OLL.

As in the the Easy stage, the OLL approach demonstrates features of high robustness since it emerges interesting games in the vast majority of cases (i.e. in 8 out of 9 cases

$I > 0.7$). In particular, in 5 out of 9 different OLL attempts the best interest value is significantly greater than or statistically equal to the respective Follower's value (i.e. 0.771 against CB, 0.772 against RB and 0.771 against ADV). It also appears that OLL is able to recover sudden disruptive mutations that cause undesired drops in the game's interest.

As previously noted also in the Easy stage OLL experiments, given an interesting initial behavior (see Figure 6.7(b)) it takes some few thousands of games for the learning mechanism to produce games of high interest. However, it takes some several thousand games to transform an uninteresting near-optimal blocking behavior (see Figure 6.7(a) and Figure 6.7(c)) into an interesting one.

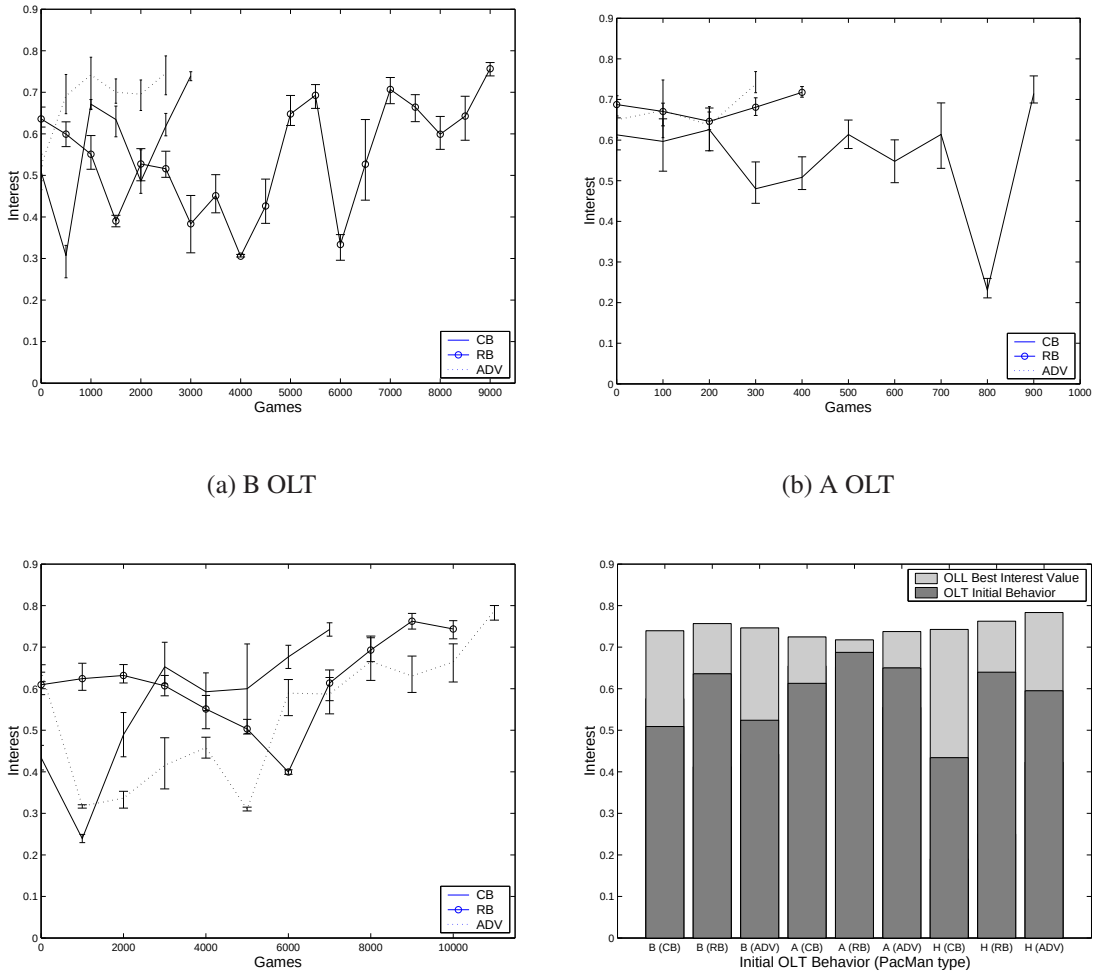
If the time required for the OLL to generate interesting games in the Normal stage is compared to the respective time in the Easy stage, there are observable differences in some learning attempts. In particular, significant time differences can be found when the initial OLT behavior is (according to the notation used in Figure 6.7(d)) B (ADV), A (ADV), H (CB) and H (RB). Since in all these learning attempts the initial interest value in the Normal stage is lower than the respective value in the Easy stage, we can assume the dependence of the convergence (in terms of interest) time on the initial interest value (which maps to a specific opponent behavior).

6.8.4 Hard Stage

The most complex stage of this version of Pac-Man, that is the Hard stage, is devised as the extreme complexity scenario to further test the OLL mechanism' ability to generate interesting opponents to play against. The OLL experiment presented here follows the steps described previously for the Easy B and Normal stages in Section 6.8.2 and Section 6.8.3 respectively.

The experiment's outcome for the Hard Stage is illustrated in Figure 6.8 where the evolution of interest over the OLL games of each one of the three different OLT behaviors is presented in each of its sub-figures. Figure 6.8(d) illustrates the overall picture of the OLL experiments by comparing the initial interest of the game against the best average interest value achieved from OLL.

As seen from Figure 6.8, the highly robust OLL approach generates interesting games in all nine OLL attempts (i.e. $I > 0.7$). More comprehensively, the best interest value is



(a) B OLT

(b) A OLT

(c) H OLT

(d) Aggregate experiment picture

Figure 6.8: Hard stage: Game interest over the number of OLL games. For reasons of computational effort, the OLL procedure continues for a number of games large enough to illustrate its behavior, after a game of high interest ($I \geq 0.7$) is found. Initial *Ghost* behaviors appear in (a), (b) and (c) sub-figure captions whereas (d) illustrates the overall picture of the experiment. Experiment Parameters: $e_p = 25$ simulation steps, $p_m = 0.02$, 5-hidden neurons controller.

significantly greater or statistically equal to the respective Follower's value (i.e. 0.772 against CB, 0.754 against RB and 0.762 against ADV) in three out of nine different OLL attempts.

It appears that given an interesting initial behavior (see Figure 6.8(b)) it only takes some few hundreds of games for the learning mechanism to produce games of high

interest, which proves to be much faster than all stages previously examined. However, it takes some several thousand games to transform an uninteresting near-optimal blocking behavior (see Figure 6.8(a) and Figure 6.8(c)) into an interesting one.

In comparison to the other stages used, some learning attempts in the Hard stage achieve relatively faster convergence times (towards the generation of highly interesting games) even though they are applied in the most complex stage. As also noticed in the Normal stage, the initial interest value of such OLL attempts in the Hard stage appears to be relatively higher than the respective values of the Normal and the Easy stage. Such an observed effect reinforces the evidence that convergence time is dependent on the initial opponent behavior (see Figure D.2 in Appendix D for a scatter plot of the initial I values over the convergence time).

6.8.5 How Does OLL Work in Pac-Man?

In Section 5.9.1 we discussed potential reasons for the success of the OLL approach in Dead End. The OLL conceptual features that generate highly interesting games are identical for the Pac-Man game and will therefore not be discussed further. However, in this section we will attempt to experimentally investigate our hypotheses.

Figure 6.9 illustrates the dependencies between the T, S, $E\{H_n\}$ and I values over on-line learning games. In the specific experiment, we let a group of B *Ghosts* to play against the ADV *PacMan* in the Normal stage and we record the aforementioned values every 100 games. As seen from Figure 6.9, OLL initially increases the *Ghosts*' spatial diversity ($E\{H_n\}$) which furthermore produces more appropriate challenge for the player (T) through the player-opponent interaction. As in Dead End, the game reaches its highest interest when the player discovers new ways of playing that the opponents can counter (increase of behavior diversity — S).

Additional evidence of the OLL approach's behavior is presented in Figure 6.10 where the correlation coefficients (r_c) between T, S and $E\{H_n\}$ values over the I value intervals are illustrated (see also Figure D.3 in Appendix D). A number of instances of these values (128) is obtained from three OLL experiments (B, A and H initial *Ghosts*) against the ADV *PacMan* in the Normal stage. According to Figure 6.10, when $I < 0.6$, T and S are highly correlated and $E\{H_n\}$ is highly anticorrelated with both T and S. When $0.6 \leq I < 0.7$, T and S are slightly anticorrelated ($r_c = -0.0974$) and $E\{H_n\}$

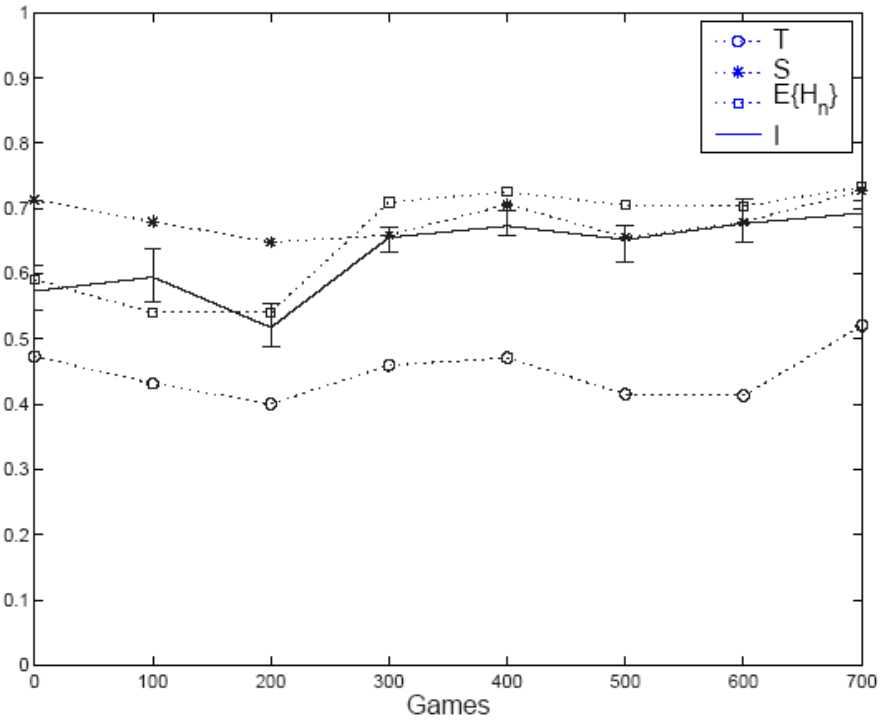


Figure 6.9: Comparison figure of T, S, $E\{H_n\}$ and I. Initial opponent behavior: Blocking.

is highly correlated and anticorrelated with T ($r_c = 0.5247$) and S ($r_c = -0.7296$) respectively. Finally, when $I \geq 0.7$, all three interest criteria are highly correlated. These correlation coefficients denote that high spatial diversity is likely to produce higher challenge (when $I \geq 0.6$) and furthermore higher *Ghosts*' behavior diversity when $I \geq 0.7$.

6.8.6 Summary

There is already much evidence that the OLL mechanism is able to find ways of increasing the interest of the game regardless of the stage complexity, topology, initial OLT *Ghost* behavior and *PacMan* type. The robustness of the mechanism is demonstrated through the fact that, starting from suboptimal OLT *Ghosts*, against any *PacMan* type in any stage it manages to emerge interesting games in the vast majority of OLL attempts. Moreover, in nearly all cases, the interest measure is kept at the same level independently of stage complexity, stage topology, player type and initial behavior. Given the confidence intervals (± 0.0537 maximum, ± 0.0238 on average — see Table D.2 in Appendix D) of the best interest values, it is revealed that the emergent

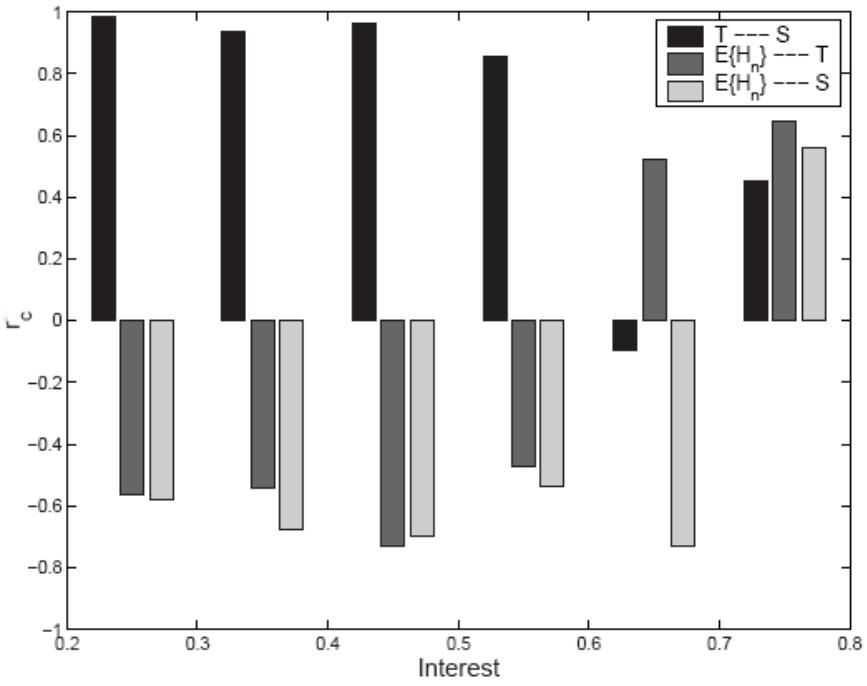


Figure 6.10: Correlation coefficients of OLL generated T, S and $E\{H_n\}$ values over I value intervals.

interest is not significantly different from stage to stage and player to player — see also Figure 6.11.

However, the evolution of the interest value over the on-line learning games appears quite erratic in most of the learning attempts. Even though the OLL mechanism includes a pre-evaluation method (see Step 4 of the algorithm presented in Section 6.6) to prevent undesired mutations, poor evaluation of the mutated *Ghost*'s behavior might lead to sudden drops of the I value. Likewise, a well-behaved mutant may boost the interest's value. The primary explanation for this noisy behavior is the number of opponents in the game. Being only four, each *Ghost*'s contribution in the interest value of the game becomes highly significant. However, as seen from the obtained results, such erratic phenomena are unlikely to happen when the interest of the game is high.

It is obvious that a number in the scale of 10^3 constitutes an unrealistic number of games for a human player to play. On that basis, it is very unlikely for a human to play so many games in order to notice the game's interest increasing. The reasons for the OLL process being that slow is its time dependence on the initial OLT behavior (see Figure D.2 in Appendix D) and a matter of keeping an appropriate balance between

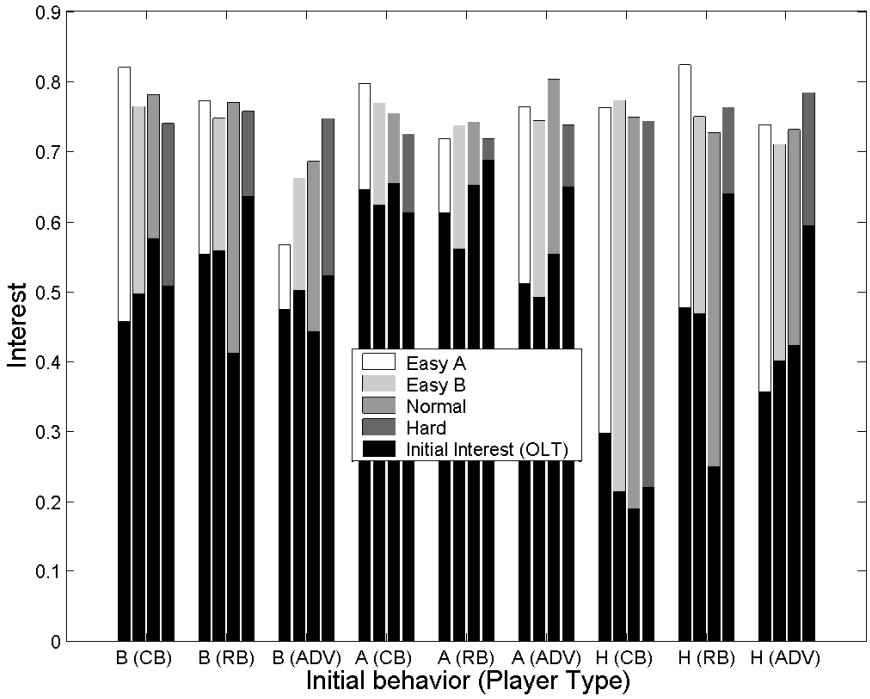


Figure 6.11: On-line learning effect on the interest of the game. Best interest values achieved from on-line learning on *Ghosts* trained off-line (B, A, H). Experiment Parameters: $e_p = 25$ simulation steps, $p_m = 0.02$, 5-hidden neurons controller.

the process' speed and its 'smoothness' (by 'smoothness' we define the interest's magnitude of change over the games). A solution to this problem is to consider the initial long period of disruption as an off-line learning procedure and start playing as soon as the game's interest is increased. How effective will this mechanism be in a potential change from a fixed strategy to a human *PacMan* player? Section 6.9 provides evidence in order to support the answer.

6.9 Adaptability Experiments

When OLL was tested in the Easy A stage in Section 6.8.1 it turned out that the mechanism was able to adapt to new — unknown during off-line training — playing strategies and enhance the player's entertainment. Additional experiments are held here in order to further support the hypothesis of the mechanism's adaptability.

In order to test the OLL approach's ability to adapt to a changing environment (i.e. change of *PacMan* strategy), the following experiment is proposed. Beginning from

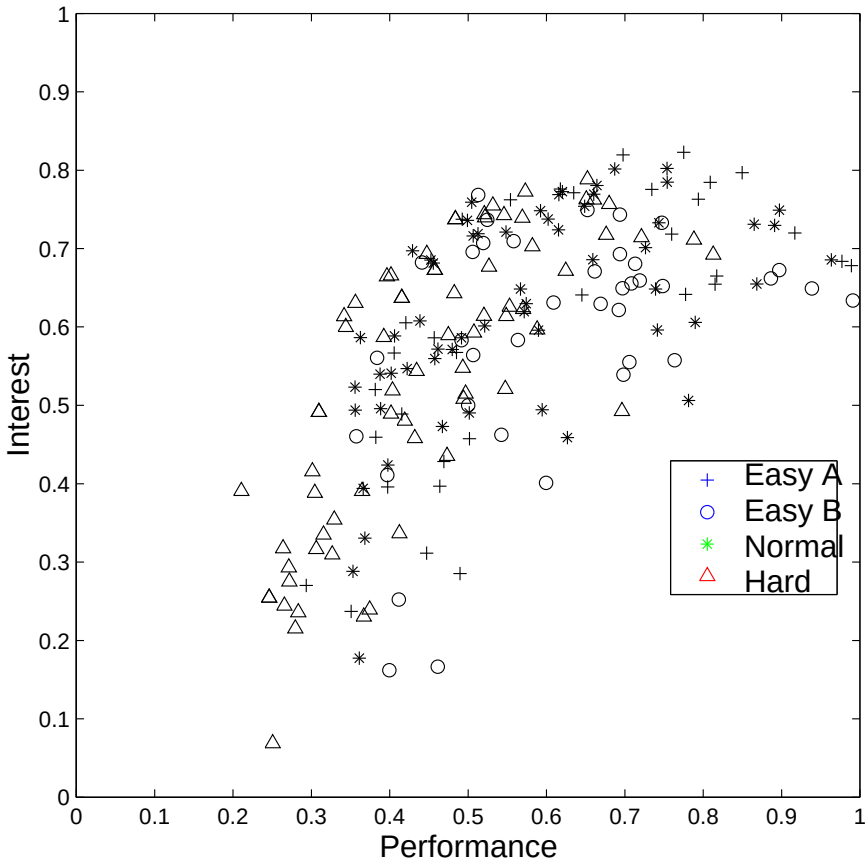
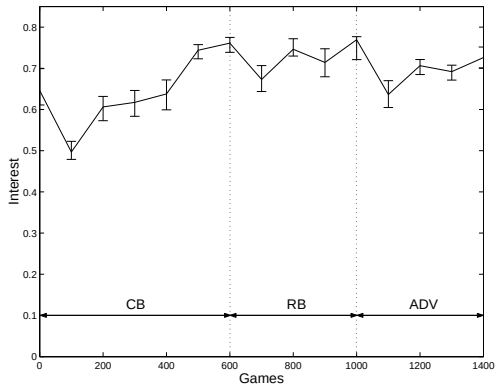


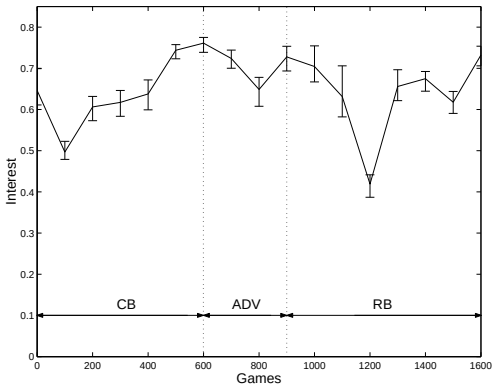
Figure 6.12: Scatter plot of I and P value instances for all four Pac-Man stages.

an initial behavior of high interest value I_{init} we apply the OLL mechanism against a specific *PacMan* type. During the on-line process we keep changing the type of player as soon as interesting games (i.e. $I \geq I_{init}$) are produced. The process stops when all three types of players have played the game. Results presented here are obtained from experiments in the Easy A, Normal and Hard stage.

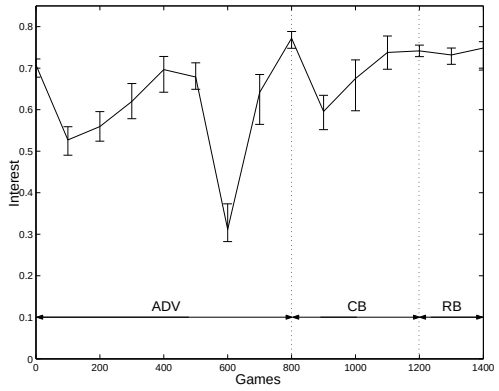
Since we have three types of players, the total number of different such experiments is 6 (all different player type sequences) for each stage. These experiments illustrate the overall picture of the approach's behavior against any sequence of *PacMan* types. As seen in Figure 6.13, Figure 6.14 and Figure 6.15, OLL is able to quickly recover a sudden change in the player's strategy and boost the game's interest at high levels after sufficient games have been played. The mechanism demonstrates a similar adaptive behavior for all 6 different sequences of *PacMan* players which illustrates its independence of the sequence of the changing *PacMan* type. Moreover, OLL adapts to new playing strategies independently of the game stage.



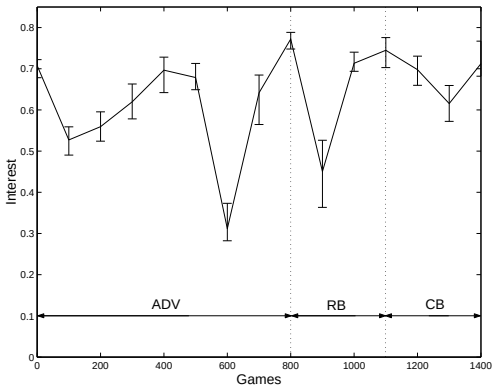
(a) CB-RB-ADV



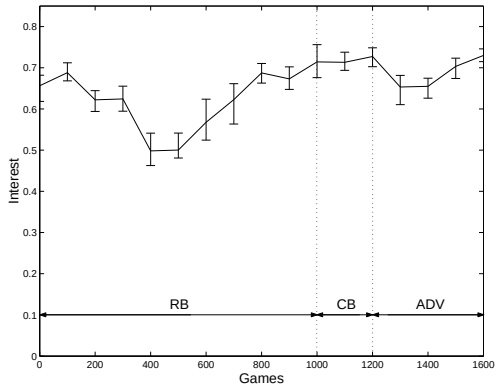
(b) CB-ADV-RB



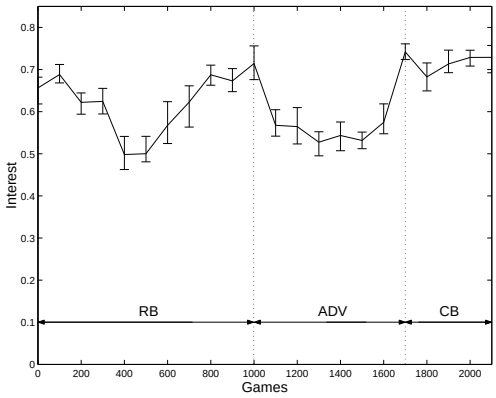
(c) ADV-CB-RB



(d) ADV-RB-CB

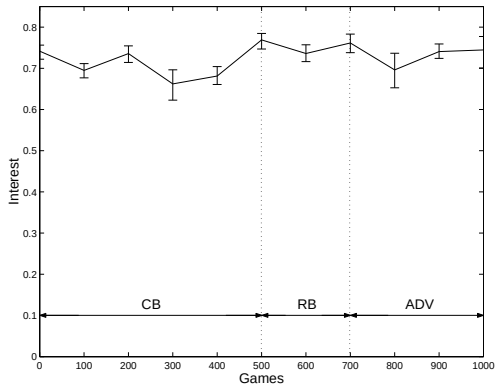


(e) RB-CB-ADV

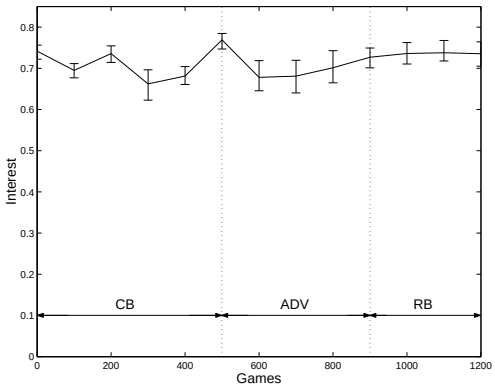


(f) RB-ADV-CB

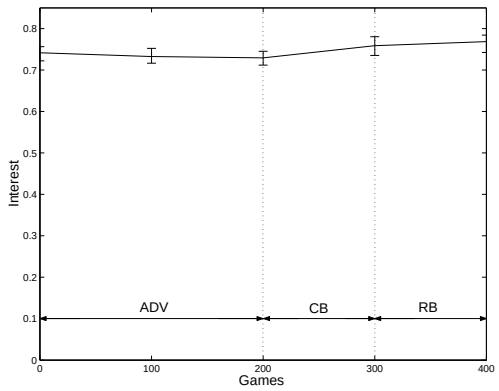
Figure 6.13: Easy A stage: On-line learning *Ghosts* playing against changing types of *PacMan*. Sub-figure captions indicate the playing *PacMan* sequence.



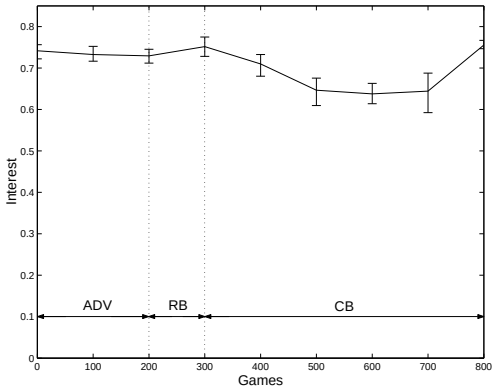
(a) CB-RB-ADV



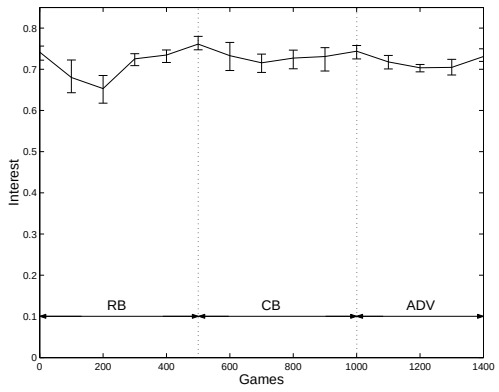
(b) CB-ADV-RB



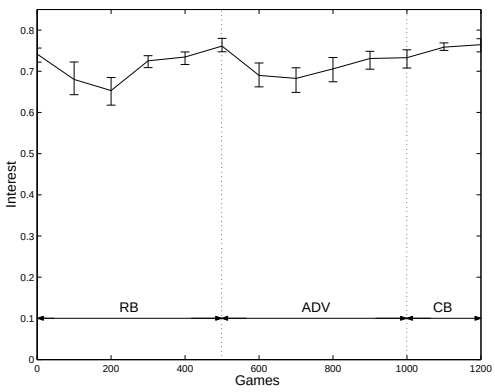
(c) ADV-CB-RB



(d) ADV-RB-CB

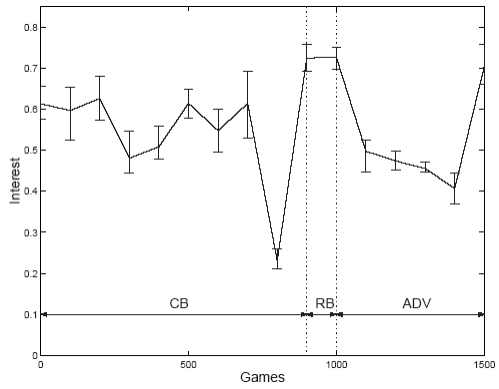


(e) RB-CB-ADV

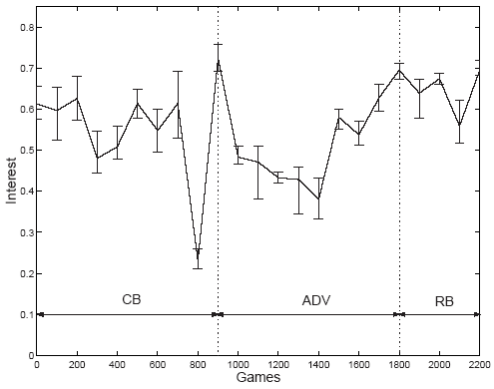


(f) RB-ADV-CB

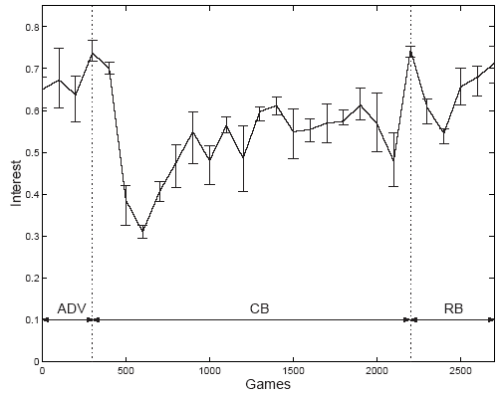
Figure 6.14: Normal stage: On-line learning *Ghosts* playing against changing types of *PacMan*. Sub-figure captions indicate the playing *PacMan* sequence.



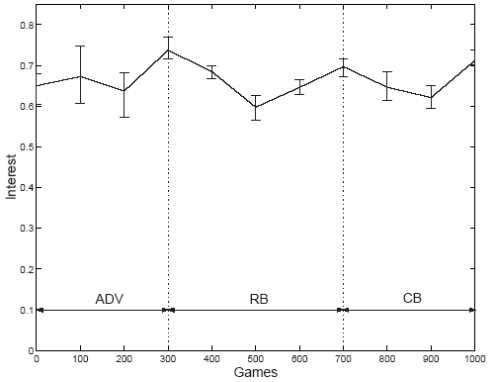
(a) CB-RB-ADV



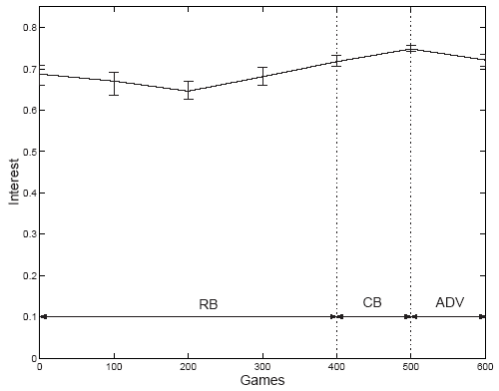
(b) CB-ADV-RB



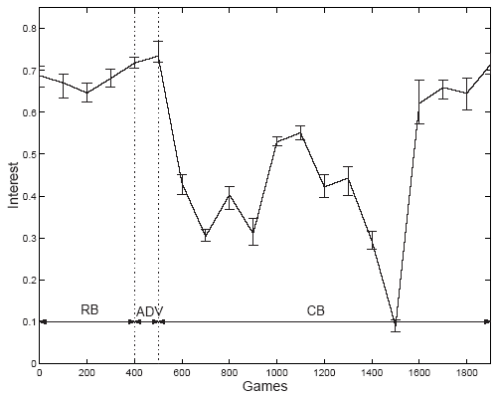
(c) ADV-CB-RB



(d) ADV-RB-CB



(e) RB-CB-ADV



(f) RB-ADV-CB

Figure 6.15: Hard stage: On-line learning *Ghosts* playing against changing types of *PacMan*. Sub-figure captions indicate the playing *PacMan* sequence.

Experiments presented here are consistent with our previous observations that convergence time is dependent on the initial interest of the game (see Figure D.2). Thus, it appears that OLL adapts to a new player faster in the Normal stage (i.e. 100 to 500 games) than in the Hard (i.e. 100 to 1500 games) and the Easy A (i.e. 200 to 1000 games) stages, since in the majority of the OLL scenarios the selected initial opponents generate a greater I value.

Results obtained from this experiment provide evidence for the approach's ability to adapt to new types of players as well as its efficiency in producing interesting games against human players. Further evidence for this hypothesis may be found in Chapter 7.

6.10 Conclusions

Given some criteria for defining interest in predator/prey games presented in Chapter 2 we introduced a generic method for measuring interest in the Pac-Man game. As in the Dead End game (see Section 5.9.1), the OLL mechanism maximizes the game's interest by rewarding aggressiveness individually (6.7). Apart from being fairly robust, the proposed approach demonstrates high and fast adaptability to changing types of player (i.e. playing strategies). Results obtained against fixed strategy *PacMan* players showed that such a mechanism could be able to produce interesting interactive opponents (i.e. games) against human playing strategies.

This chapter concludes with a discussion on the proposed methodology for obtaining predator/prey games of high interest by outlining a summary of its demonstrated generality over various dimensions of such games.

6.11 Pac-Man versus Dead End

On-line learning procedures have been successful in both games applied. In this section we further discuss, analyze and compare the games, the OLL variants, the players and the initial behaviors used.

6.11.1 Game Variants

We used two predator/prey games differing in the characters' motion type, stage environment and player objectives. Interest was able to reach high values and the OLL mechanism demonstrated robustness and adaptability when applied to both Pac-Man and Dead End games. However, no effective comparison between the two games' generated interest values can be derived and therefore no answer can be given to which game is more interesting by design. Based on the games' main features, we believe that these two test-beds cover a large portion of the properties met in the predator/prey computer game genre.

Conceptually, the primary dissimilarities between Pac-Man and Dead End are found in:

- **The player's objectives:** In Dead End the *Cat* has to avoid the *Dogs* in order to escape through the Exit whereas in Pac-Man the player has to avoid the *Ghosts* while eating pellets appearing on the stage.
- **The type of opponent motion:** In both games the movement directions are limited to up, down, left or right but while the *Ghosts*' magnitude of motion is discrete (measured in grid cells) the Dead End opponents' magnitude of motion is continuous. Moreover, the ratio of the player's over the opponents' maximum speed is 4/3 and 2 for the games of Dead End and Pac-Man respectively.
- **Walls:** The absence of objects (walls) in Dead End versus the existence of corridors in Pac-Man.

6.11.2 OLL Variants

Regarding the OLL approach variants used for the two games, experiments project a smooth but rather slow change of the interest value in Pac-Man whereas in Dead End a quite noisy (unstable) but relatively fast change is noticed. Besides, the more complex the Dead End game environment (i.e fewer *Dogs*) is, the more distinctive this instability becomes. The aforementioned dissimilarity in the interest value evolution is fully determined by (a) the pre-evaluation procedure and (b) the gaussian mutation operator that exist in the Pac-Man version of the OLL mechanism. According to the former, the probability of disruptive phenomena caused by unsuccessful mutations is

minimized and, therefore, evolution is decelerated for the sake of smooth changes in emergent behaviors (see also Chapter 9 for a discussion on this issue). According to the latter, the variance of the gaussian mutation is inversely proportional to the entropy of a group of *Ghosts*. Hence, the higher the *Ghosts*' cell visit entropy, the less disruptive the mutation process.

If we attempt to compare the two mechanisms used, the Pac-Man OLL variant appears as a more sophisticated algorithm that is designed to skip undesired opponent behaviors and erratic changes of the I value with the cost of convergence time (see also Section 9.2.1.1). On the contrary, the Dead End OLL variant is a hill-climber that generates interesting games faster with the cost of instability.

In the Dead End game, generations instead of games (as in Pac-Man) are picked as the algorithm's simulation time unit because of the large difference on the average game length between the EA *Cat* (i.e. $t_{max} = 10$) and the RM and PFB *Cat* (i.e. $t_{max} = 50$). For comparison purposes to the OLL experiments in the Pac-Man game, the expected value of the generations g per games G ratio $E\{g/G\}$ is calculated. This equals 0.943 for the RM and PFB *Cat* and 3.846 for the EA *Cat*.

6.11.3 Game Complexity

The OLL's ability to generate interest was tested over the game's complexity which corresponds to the Easy, Normal and Hard stages for Pac-Man and to five, four and three *Dogs* environments for Dead End. Results obtained from these experiments demonstrate the approach's generality since interesting games emerge independently of game complexity.

6.11.3.1 Stage Topology

In addition to stages of different complexity, topologically different Pac-Man stages of equal complexity (Easy A and Easy B) were used as test-beds for the approach. Obtained interest values showed that the topology of the stage does not seem to hinder the OLL's adaptive features surfacing.

6.11.4 Players

As far as the playing strategy is concerned, three hand-crafted players have been designed for each game. Independently of playing strategy, the mechanism adapted to their playing style in order to boost the game's interest. More specifically, in the Pac-Man game, the best OLL generated interest values were not significantly different regardless of the player type.

According to the overall observed behavior of the I value it appears that, in addition to the opponent, the player may also play a significant role for its own entertainment. The player may determine the game's plot to a degree and this occurs due to its interaction with the opponents. In that sense, the interest value is affected by the player more in extreme scenarios such as unacceptably low (i.e. unable to control the player and sense the features of the game environment) or expert (i.e. unbeatable) gaming skills (see Section 2.6). The entertainment perceived by such a player is rather low which accordingly confirms the first interest criterion of challenge in computer games (see Chapter 2).

For instance, both EA and RM (in a lesser degree) types of *Cat* belong to this category of game-playing by following a trivial low-quality game strategy. Contrary to the PFB *Cat* and all *PacMan* types used, the aforementioned player types do not interact with their opponents, which furthermore, leads to a poor estimation of the game's I value. Even though such playing strategies are rare among humans they constitute a limitation of the interest value estimation that is further discussed in Chapter 9.

6.11.5 Initial Opponents

Five different behaviors emerged from off-line training procedures were selected as initial points in the search for more interesting games. For the Pac-Man game we categorized OLT emerged behaviors into blocking, aggressive and hybrid whereas for the Dead End game the OLT behaviors obtained were characterized as either aggressive or defensive. Given these diverse initial behaviors, the OLL mechanism exhibited high robustness and fast adaptability in increasing the game's interest. Moreover, results showed that convergence time of highly interesting games is dependent on the initial interest value.

In the next chapter, we present experiments with human Pac-Man players. Given the

above-mentioned observations of the convergence time and initial opponent dependence, some quite interesting *Ghosts* are the initial opponents seeded in the on-line learning experiment against humans.

Chapter 7

Human Survey

> "*I understand the behavior of the ghosts and I am able to manipulate the ghosts into any corner of the board I choose. This allows me to clear the screen with no patterns. I chose to do it this way because I wanted to demonstrate the depths of my abilities. I wanted to raise the bar higher — to a level that no one else could match.*"
>
> Billy Mitchell, Pac-Man world champion.

Experiments against computer-programmed fixed playing strategies portrayed the OLL mechanism's ability to generate interesting predator/prey games independently of game concept, complexity, opponent and player behavior. Apart from being fairly robust, the proposed mechanism demonstrated high and fast adaptability to changing types of player (i.e. playing strategies) in both games tested. The subsequent obvious step to take is to let humans judge whether generated games are realistically interesting or not and whether OLL indeed enhances the level of entertainment during play. For this, we conducted a survey, with human subjects as *PacMan* players, that primarily aims to obtain answers to the following questions:

1. Does the interest value computed for a game correlate with human judgement of interest?

2. Does the on-line learning mechanism cause perceived interest to change? Do perceived changes match computed ones?

The experiment is comprehensively described in Section 7.1. In Section 7.2 and Section 7.3 the statistical method used and the analysis of obtained results are presented

respectively.

7.1 Experiment Description

Answers to the key questions previously presented are based on statistical analysis of data acquired from a questionnaire (see Appendix E for the questionnaire used) applied for the Pac-Man game. The main prerequisite for a subject to participate in this experiment is to have played the original version (Namco) of the Pac-Man game at least once. For this experiment, the Normal (see Figure 6.1(c)) stage is used, being the one that covers the greatest range of Pac-Man playing skills among the stages used. Hard stage is too difficult for beginners whereas Easy is too simple for highly-skilled players. The number of subjects used was thirty and their age covered a range between 17 and 51 years, where both sexes were almost equally represented (43% females, 56% males). In addition, all subjects speak English as a foreign language since their nationality was either Danish (90%) or Greek (10%). The questionnaire is divided into 3 parts (A, B and C) and the steps that the subjects went through for each part are presented as follows.

7.1.1 Part A: Personal Data

A.1 Subjects are asked to define their interest in computer games in general. The categorization is as follows: a) I love computer games; b) I like them, but I'm not that enthusiastic about them; c) I don't like computer games.

A.2 Subjects are asked to define their interest in the Pac-Man game before they play it. There are five different answer-options to choose from, which categorize participants into three different types of Pac-Man player. The options are:

1. I'm a fanatic Pac-Man player.
2. I like Pac-Man.
3. I like Pac-Man, but I am not that enthusiastic about it.
4. I used to like Pac-Man, but not any more.
5. I don't like Pac-Man.

Subjects choosing 1–2, 3 and 4–5 are assigned to the first (represented as "Like"), second (represented as "Neutral") and third (represented as "Don't like") type of player respectively.

A.3 Subjects are asked to list the factors they consider make a better Pac-Man game. Data from this answer are used for correlation with answers to question C.2 (see Section 7.1.3).

A.4 Subjects familiarize themselves with the game by playing 50 games against specific OLT opponents (i.e. opponent 4 presented in Table 7.1). On-line learning is used during this testing period, which is not noticeable to the player. At the end of the testing period, each subject's opponents trained on-line are saved.

7.1.2 Part B: 1^{st} Objective

We pick opponents differing in the interest value generated when playing against the ADV player (as the most advanced computer-guided *PacMan* player). We select five opponents whose computed interest values uniformly cover the $[0,1]$ space. The selected opponent's numbers, which are used as id-codes, and their respective interest values, are presented in Table 7.1.

By experimental design, each subject plays against three of the selected opponents in all permutations of pairs. In addition, we require equal participation of all three player types. For this experiment, we use thirty subjects divided into three equal subsets for each of the three player types (Like, Neutral, Don't Like), since $C_3^5 = 10$ — all combinations of 3 out of 5 opponents — subjects are required for each player type. Moreover, observed effects show that thirty subjects constitute a statistically significant sample (see section 7.3).

B.1 As previously mentioned, each subject plays sets of games (five games in each set) against three of the selected opponents in all permutations of pairs and each time a pair of sets is completed, the player is asked whether the first set was more interesting than the second set of games.

The total number of different sets of games that is played by each subject is twelve (all permutations of three pairs — e.g. if 1, 2 and 3 are selected then the subject plays the following six pairs of sets: $[1,2]$, $[2,1]$, $[1,3]$, $[3,1]$, $[2,3]$, $[3,2]$). The sequence

Opponent	I_u	I	I_l
1	0.2043	0.1793	0.1494
2	0.3673	0.3158	0.2670
3	0.5501	0.4943	0.4420
4	0.6706	0.6484	0.6267
5	0.8180	0.8023	0.7858

Table 7.1: The selected opponents and their respective interest — I and 95% confidence interval (I_u, I_l) values.

of the six pairs of sets that each subject plays is defined *a priori* given the following conditions (see Table E.3 in Appendix E):

(1) Each pair [A–B] is played in a different place of the six pair sequence each time it is played and

(2) No [A–B] pair is adjacent to [B–A] pair. This way we minimize the effect of the pairs' playing order.

Given thirty subjects, there are nine observed incidents for each pair of sets.

7.1.3 Part C: 2^{nd} Objective

C.1 Each subject plays 25 games against the initial training phase opponents (i.e. opponent 4 — OP4) and 25 games against the on-line trained opponents that were saved (i.e. two sets of games). We let each subject play another two sets against these opponents in different order. Half of the subjects play these four sets of games in the sequence OLL-OP4, OP4-OLL, whereas the other half play them in the sequence OP4-OLL, OLL-OP4 since we require minimization of any potential ordering effect. Each time a pair of sets (two pairs here) is finished, the player is asked whether the first set was more interesting than the second set of games.

In order to calculate the interest value for each of the 2 sets, we record the e (pellets eaten), K (*PacMan* kills), t_k (time to kill *PacMan*) and v_{ik} (total number of the opponents' cell visits) values while subjects play, obtaining data of 50 games against each opponent in total (see Section 7.3.4).

C.2 Subjects are asked to list the criteria they used for their assessment of which set of games was more interesting. This last question is added to cross-check (along with question A.3 before the testing period) if subjects' factors of a good Pac-Man game before playing correlate with the criteria of assessment of an interesting Pac-Man game after playing.

7.2 Method

Hypothesis testing is the use of statistics to determine the probability that a given hypothesis is true or false. The usual process of hypothesis testing consists of four steps.

1. Formulate the null hypotheses (see Section 7.2.1).
2. Identify a test statistic that can be used to assess the truth of each null hypothesis (see Section 7.2.2).
3. Compute the probability that each test statistic would assume a value greater than or equal to the observed value strictly by chance, called the p-value (see Section 7.2.3).
4. Compare the obtained p-values to an acceptable significance value (see Section 7.2.4).

7.2.1 Hypotheses

For this experiment there are three null hypotheses formed:

H_0: The correlation between observed human judgement of interest and the computed interest value, as far as the different opponents are concerned, is a result of randomness.

H_1: Observed human judgement of interest does not correlate with the computed interest value, as far as the different opponents are concerned.

H_2: Observed human judgement of interest does not correlate with performance during play.

7.2.2 Test Statistic

Given the interest metric (2.5) and two sets of games A and B, it can be determined that "game A is more (or less) interesting than game B". In answer to the same question, a human subject can indicate that either $I_A > I_B$ or $I_A < I_B$. In order to measure the degree of agreement between the human judgement of interest and the interest value given by (2.5), we calculate the correlation coefficients

$$c(\overrightarrow{z}) = \sum_{i=1}^{\mathbf{N}} \frac{z_i}{\mathbf{N}} \tag{7.1}$$

where $\mathbf{N}$ is the number of incidents to correlate and

$$\overrightarrow{z} = \begin{cases} 1, & \text{if subject agrees with (2.5);} \\ -1, & \text{if subject disagrees with (2.5).} \end{cases} \tag{7.2}$$

The test statistic (7.1) is used to assess the truth of all three null hypotheses. However, for the null hypothesis H_2, the correlation coefficients $c(\overrightarrow{z'})$ are computed where z' values are obtained from (7.3).

$$\overrightarrow{z'} = \begin{cases} 1, & \text{if subject chooses according to performance;} \\ -1, & \text{if subject does not choose according to performance.} \end{cases} \tag{7.3}$$

7.2.3 P-values

The p-value for this experiment is the probability $P(C \geq c)$ that a correlation coefficient C at least as significant as the one observed c, would be obtained assuming that the null hypothesis was true. Thus, the smaller this probability, the stronger the evidence against the null hypothesis. The distribution used for obtaining the correlation coefficient probabilities is the Binomial (7.4).

$$P(n) = \frac{\mathbf{N}!}{n!(\mathbf{N}-n)!} p^n (1-p)^{(\mathbf{N}-n)} \tag{7.4}$$

where $p = 1/2$ and $n = [\mathbf{N}(c+1)]/2$. For $\mathbf{N} \to \infty$ the Binomial distributed correlation coefficient can be approximated by the Normal distribution (7.5).

$$c \rightsquigarrow \mathcal{N}(\mathbf{N}p, \sqrt{\mathbf{N}p(1-p)}) \tag{7.5}$$

For the experiments presented here we use the Normal distribution approximation for $N > 90$.

7.2.4 Significance

Generally if the p-value of an experiment is less than or equal to a significance value ε ($P(C \geq c) \leq \varepsilon$) the observed effect is statistically significant and the null hypothesis is ruled out. For the experiments presented here, if $P(C \geq c) \leq 1\%$ then the observed effect is "highly significant", if $1\% < P(C \geq c) \leq 5\%$ then the observed effect is "significant" and if $P(C \geq c) > 5\%$ then the observed effect is "not significant".

7.3 Statistical Analysis

As noted in Chapter 2, this work concentrates on the characters' behavioral aspect of interesting games. More specifically, it focuses on the opponent's rather than the graphics' or the sound's impact on the player's entertainment. Apart from the opponent, there are two additional factors that may affect the interest of a computer game, that are examined in this section. These are the player-subject type (degree of *a priori* game liking) and the order of play.

7.3.1 Opponent

Each entity in Table 7.2 represents a subject's answer to the question B.1, equivalent to "Is $I_i > I_j$?", where i, j the row and column number respectively. Given the interest values of the five opponents (see Table 7.1), 'O' and 'X' stand respectively for the subject's agreement and disagreement with this ranking (in other words, O and X characters are selected for visual purposes to symbolize the respective z values — see (7.2)). As stressed before, given thirty subjects, there are nine incidents for each pair of opponents which are represented in a 3×3 matrix. Rows within this matrix denote the type of the subject that answered the specific question. In particular, traversing from the top to the bottom row of the matrix, the liking alters from 'Like' to 'Neutral' and finally to 'Don't Like'.

	Is $I_{Row} > I_{Column}$?														
	1			2			3			4			5		
1				O	O	X	O	O	X	O	O	X	O	O	O
				O	O	X	O	O	O	O	O	X	O	O	X
				O	O	O	O	O	O	O	O	O	O	X	X
2	X	X	X				O	O	O	O	O	X	O	O	O
	X	X	X				O	O	X	O	O	O	O	O	X
	O	X	X				O	O	O	O	O	X	O	O	X
3	O	X	X	O	O	O				O	X	X	O	O	X
	O	O	X	O	O	X				O	X	X	O	O	O
	O	X	X	O	O	X				O	O	X	O	O	X
4	O	O	X	O	O	O	X	X	X				O	O	X
	O	O	X	O	O	X	X	X	X				O	O	X
	O	X	X	O	O	X	O	X	X				O	O	O
5	O	O	X	O	O	O	O	O	O	O	O	O			
	O	O	O	O	O	X	O	O	X	O	O	O			
	O	O	O	O	O	O	O	O	X	O	O	O			

Table 7.2: Agreement between the subject's judgement of interest and the interest metric — O: $z = 1$, X: $z = -1$.

Table 7.3 presents the correlation coefficients and their respective $P(C \geq c)$ values for each one of the ten combinations of opponent pairs ($\mathbf{N} = 18$) and in total ($\mathbf{N} = 180$). There is an obvious disagreement between the interest metric and the human's notion of interest in opponent pairs 1–2 and 3–4. Even though humans seem to agree with the interest metric in the pairs 1–3 and 1–4, the obtained p-values reveal statistically insignificant results. For the rest of the pairs we experience statistically highly significant (i.e. 2–3, 2–5, 4–5) and significant (i.e. 1–5, 2–4, 3–5) matching to observed human judgement. Finally, the total agreement correlation coefficient $(c = 0.3888)$ as well as its p-value $(P(C \geq c) = 1.31 \cdot 10^{-7})$ demonstrate a statistically highly significant effect that rules out the null hypothesis H_1. Thus, it appears that the observed human judgement of interest correlates with the computed interest value, as far as the different opponents are concerned. Moreover, the obtained p-values presented in Table 7.3 illustrate that the sample size of thirty subjects is adequate to produce statistically significant observed effects.

Pair	c	$P(C \geq c)$
1–2	−0.1111	0.7596
1–3	0.3333	0.1189
1–4	0.3333	0.1189
1–5	0.5555	0.0154
2–3	0.6666	0.0037
2–4	0.5555	0.0154
2–5	0.7777	0.0006
3–4	−0.4444	0.9845
3–5	0.5555	0.0154
4–5	0.6666	0.0037
Total	0.3888	$1.31 \cdot 10^{-7}$

Table 7.3: Interest metric - Subject judgement correlation coefficients c and $P(C \geq c)$ values for all pairs of opponents and in total.

Opponent	*PacMan* Type	I_u	I	I_l
1	CB	0.3307	0.3026	0.2447
	RB	0.4397	0.4147	0.3931
	ADV	0.2043	0.1793	0.1494
2	ADV	0.3673	0.3158	0.2670

Table 7.4: Generated interest of opponent 1 against all *PacMan* types — I and 95% confidence interval (I_u, I_l) values.

7.3.1.1 Opponent 1

Further investigation of the interest value generated by opponent 1 showed high dependence on the player type. More specifically, when opponent 1 plays against the CB *PacMan* and the RB *PacMan*, it generates interest which is respectively statistically not different and significantly higher than the interest generated by opponent 2 (see Table 7.4). Opponent 1 constitutes a particular case since no such change in the opponent ranking (i.e. ranked by interest) occurs for any other of the four remaining opponents.

Given the ranking instability of opponent 1, we recalculate the z values as if 1) $I_1 > I_2$

Pair	c	$P(C \geq c)$
(1,2)–3	0.5000	0.0019
(1,2)–4	0.4444	0.0056
(1,2)–5	0.6667	$3.5 \cdot 10^{-5}$
3–4	−0.4444	0.9845
3–5	0.5555	0.0154
4–5	0.6666	0.0037
Total	0.4444	$1.17 \cdot 10^{-8}$

Table 7.5: Interest metric - Subject judgement correlation coefficients c and $P(C \geq c)$ values when $I_1 = I_2$ is assumed.

and 2) $I_1 = I_2$ and proceed as above. In the first case, the z values of the [1–2] pair swap their sign and the obtained p-values for this pair and in total are 0.4072 and $2.57 \cdot 10^{-8}$ respectively. For the latter case, the z values of the [1–2] pair are not taken into consideration and the two first (triplets of) rows and columns of Table 7.2 are merged into one by adding up their z values. The obtained p-values for the remaining six pairs and in total are presented in Table 7.5. For both cases, changes in the opponent 1 ranking increase the significance of the observed effects.

7.3.2 Subject Type

In this section we present how the subject's type, which corresponds to the subject's "liking of the Pac-Man game", correlates with the subject's judgement of interest. To this end we compute the correlation coefficients c and their respective probabilities $P(C \geq c)$ for each subject type (60 incidents for each type).

As seen from Table 7.6, all three types of subject's observed judgement of interest collectively demonstrate a highly significant agreement ($P < 1\%$) with the interest metric. However, it appears that there is no significant difference between the three different types and, therefore, no secure conclusions about the subject's type effect on its notion of interest can be arisen.

Subject Type	c	σ_c^2	$P(C \geq c)$
Like	0.4000	0.0691	0.0013
Neutral	0.3333	0.1234	0.0067
Don't like	0.4333	0.1493	0.0005
Total	0.3888	0.1079	$1.31 \cdot 10^{-7}$

Table 7.6: Interest metric - Subject judgement correlation coefficients c, $P(C \geq c)$ values of the three different types of subject and correlation variance (σ_c^2) over the 10 subjects of each type.

	$I_1 > I_2$		$I_1 = I_2$	
Subject Type	c	$P(C \geq c)$	c	$P(C \geq c)$
Like	0.4666	0.0001	0.4814	0.0002
Neutral	0.4000	0.0013	0.4074	0.0019
Don't like	0.3666	0.0031	0.4444	0.0007
Total	0.4111	$2.57 \cdot 10^{-8}$	0.4444	$1.17 \cdot 10^{-8}$

Table 7.7: Interest metric - Subject judgement correlation coefficients c and $P(C \geq c)$ values of the three different types of subject when $I_1 > I_2$ and $I_1 = I_2$.

7.3.2.1 Opponent 1

By following the procedure described in section 7.3.1.1 for the particular case of opponent 1 we also come up with highly significant values for all three subject types and no significant difference between them for both cases of $I_1 > I_2$ and $I_1 = I_2$ (see Table 7.7).

7.3.3 Order of Play

In order to check whether the order of playing Pac-Man games affects the human judgement of interest, we hypothesize that there is no order effect and proceed as follows. For each pair of opponents, that a subject played in both orders, we count a) the times K that the subject agrees with the interest value only in the first pair played and b) the times J that the subject agrees with the interest value only in the latter pair played. In the case where the subject agrees or disagrees with the interest value in both pairs played, we take no action. To this end, we compute the z'' value (7.6) for each

Pair	z''	$P(Z \geq \lvert z''\rvert)$
1–2	0.2222	0.2403
1–3	0.3333	0.1189
1–4	−0.2222	0.2403
1–5	−0.1111	0.4072
2–3	0.1111	0.4072
2–4	0.1111	0.4072
2–5	0.0000	0.5927
3–4	−0.1111	0.4072
3–5	−0.3333	0.1189
4–5	0.2222	0.2403
Total	0.0222	0.4818

Table 7.8: Order of play test statistic z'' and $P(Z \geq |z''|)$ values for all pairs of opponents.

pair of opponents ($\mathbf{N} = 9$) and in total ($\mathbf{N} = 90$).

$$z''(K,J) = (K-J)/\mathbf{N} \tag{7.6}$$

The greater the absolute value of $z''(K,J)$ the more the order of play tends to affect the subjects' judgement of interest. This value defines the test statistic used to assess the truth of the hypothesis that there is no order effect. The obtained z'' value is Trinomial distributed according to (7.7).

$$P(K,J) = \frac{\mathbf{N}!}{K!J!(\mathbf{N}-J-K)!} p^K q^J (1-p-q)^{(\mathbf{N}-J-K)} \tag{7.7}$$

where $p = q = 0.25$, giving equal probabilities to the K (agree only in the first pair played) and J (disagree only in the first pair played) events and a probability of $(1 - p - q) = 0.5$ to the event of agreeing or disagreeing in both pairs played. P-values $P(Z \geq |z''|)$ for each pair and in total are obtained by using (7.7) and presented, along with their respective z'' values, in Table 7.8.

As seen from Table 7.8 there are no statistically significant effects in any pair of opponents or in total. Therefore, the null hypothesis is not rejected and it seems that the order of play does not affect the human judgement of interest.

Pair	z''	$P(Z \geq \lvert z'' \rvert)$
(1,2)–3	0.2222	0.1214
(1,2)–4	−0.0555	0.4339
(1,2)–5	−0.0555	0.4339
3–4	−0.1111	0.4072
3–5	−0.3333	0.1189
4–5	0.2222	0.2403
Total	0.0	0.5312

Table 7.9: Order of play test statistic z'' and $P(Z \geq |z''|)$ values for all pairs of opponents when $I_1 = I_2$ is assumed — $\mathbf{N} = 36$ for the pairs (1,2)–3, (1,2)–4 and (1,2)–5.

7.3.3.1 Opponent 1

Order of play is not affected by the particular behavior of opponent 1 either. That is, if $I_1 > I_2$ there is no difference in the obtained $P(Z \geq |z''|)$ values and if $I_1 = I_2$ we also come up with no statistically significant effects in any pair of opponents or in total (i.e. $P(Z \geq |z''|) = 0.5312$, $\mathbf{N} = 81$). For a detailed reference, see Table 7.9.

7.3.4 On-Line Learning

In this section we analyze the observed effects from the on-line learning experiment (Part C) presented in Section 7.1.3. In Part C, subjects play 2 sets of 50 games in total. The bootstrapping procedure presented in Appendix A, with $N = 25$, is used to determine the *Ghosts'* average interest values against each human subject as well as its 95% confidence interval. Interest values calculated and presented in Table 7.10 show that in 18 out of 30 cases the human player managed to produce more interesting games by the use of the on-line learning procedure. However, it is not clear whether OLL used against humans cause the interest value to proliferate. Thus, it seems that 50 OLL games (testing period in Part A) are not adequate for the OLL mechanism to cause a significant difference in the interest value.

Choosing an on-line learning period (or else testing period) of 50 games is an empirical way of balancing efficiency and experimental time. The duration of the testing period lasted 20 minutes on average whereas the whole experiment exceeded 65 minutes in

	OLL			No OLL		
Subject	I_u	I_l	I	I_u	I_l	I
1	0.721	0.575	0.671	0.745	0.393	0.630
2	0.753	0.588	0.669	0.767	0.593	0.703
3	0.733	0.614	0.669	0.755	0.607	0.694
4	0.805	0.672	0.735	0.792	0.520	0.677
5	0.802	0.644	0.711	0.720	0.582	0.665
6	0.763	0.598	0.676	0.733	0.531	0.647
7	0.725	0.638	0.689	0.698	0.559	0.644
8	0.751	0.566	0.673	0.804	0.603	0.720
9	0.746	0.568	0.681	0.751	0.630	0.698
10	0.780	0.531	0.670	0.780	0.616	0.715
11	0.692	0.469	0.619	0.750	0.576	0.695
12	0.802	0.678	0.748	0.865	0.700	0.778
13	0.806	0.530	0.662	0.716	0.532	0.638
14	0.799	0.589	0.715	0.805	0.678	0.738
15	0.776	0.636	0.707	0.782	0.656	0.706
16	0.812	0.658	0.749	0.806	0.689	0.745
17	0.784	0.601	0.706	0.743	0.609	0.679
18	0.796	0.595	0.708	0.740	0.567	0.655
19	0.780	0.612	0.702	0.718	0.626	0.670
20	0.749	0.666	0.717	0.759	0.646	0.716
21	0.753	0.625	0.684	0.757	0.659	0.706
22	0.790	0.660	0.728	0.831	0.625	0.733
23	0.774	0.640	0.709	0.762	0.663	0.700
24	0.752	0.599	0.668	0.754	0.612	0.681
25	0.741	0.635	0.696	0.705	0.589	0.660
26	0.825	0.697	0.770	0.781	0.681	0.728
27	0.799	0.622	0.732	0.782	0.640	0.724
28	0.786	0.630	0.719	0.755	0.570	0.693
29	0.745	0.607	0.690	0.748	0.606	0.705
30	0.793	0.673	0.738	0.782	0.591	0.678
$E\{\}$	0.771	0.614	0.700	0.763	0.605	0.694

Table 7.10: Interest I and 95% confidence interval (I_u, I_l) values against all 30 human players ranked by subject type. I.e. 1–10: Like, 11–20: Neutral, 21–30: Don't Like.

many cases, which is a great amount of time for a human to be constantly concentrated. Fixed strategy *PacMan* player results (see Chapter 6) showed that more on-line learning games are required for the interest value to change significantly, which apparently

Liking	c	$P(C \geq c)$	z''	$P(Z \geq \lvert z'' \rvert)$
Like	0.2	0.2517	0.0	0.5881
Neutral	0.4	0.0577	0.0	0.5881
Don't like	-0.1	0.7483	0.3	0.1315
Total	0.1666	0.1225	0.1	0.2594

Table 7.11: On-line learning against humans: interest metric agreement c and order of play z'' test statistics and their respective p-values sorted by subject type.

seems to be the case for human players as well.

By calculating the correlation coefficient (7.1) between the computed interest values (presented in Table 7.10) and the human judgment of interest obtained by question C.1, we get a value of $c = 0.1666$ with corresponding probability of $P(C \geq c) = 0.1225$ for $\mathbf{N} = 60$. This does not constitute a statistically significant effect and suggests that humans were not able to tell the difference between opponent 4 and the opponents trained on-line at the end of the testing period.

Moreover, in order to check how the subject's type affects its judgement when on-line learning runs in the background, we compute the correlation coefficients (see (7.1)) and their corresponding p-values for each of the three types of subject ($\mathbf{N} = 20$). Results presented in Table 7.11 do not display a statistically significant effect from any of the three subject types.

Finally, in order to examine whether the order of playing Pac-Man, with and without on-line learning, affects the human judgement of interest we hypothesize that there is no order effect and proceed as in Section 7.3.3. Thus, we compute the z'' value (7.6) for each pair of opponents for all subjects ($\mathbf{N} = 30$). P-values $P(Z \geq \lvert z'' \rvert)$ are obtained by using (7.7) and presented, along with their respective z'' values, in Table 7.11. As in Section 7.3.3, results do not show any statistically significant effect and therefore it seems that the order of play, where on-line learning is switched on and off sequentially, does not affect human's judgement of interest.

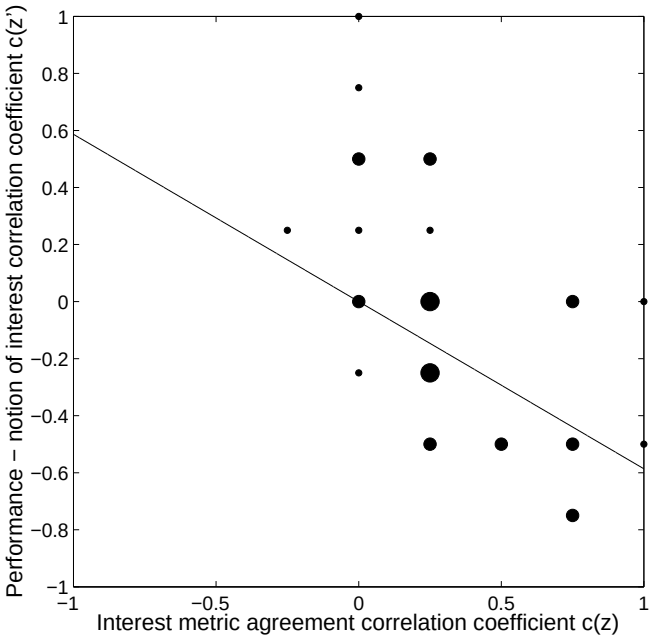


Figure 7.1: Scatter plot of $c(z)$ and $c(z')$ values for each subject and their statistical correlation's line. The circular marker's radius is increased in respect to the number of occurrences (i.e. 1, 2 or 3).

7.3.5 Performance Factor

As noted before, each subject plays eight pairs of sets of games in total during this experiment (six in Part B and two in Part C), and each set is assigned a score that corresponds to the performance of the subject. More specifically, the score is directly proportional to the number of pellets eaten by the player (see Table E.4 in Appendix E). Given the subjects' scores and the observed interest judgement obtained from questions B.1 and C.1, the z' values are computed as follows. If the subject chooses the set of games with the higher score obtained as being more interesting then the z' value is 1. Accordingly, the z' value is -1 if the subject chooses the set of games with the lower score obtained as being more interesting. By computing (7.1) for all thirty subjects ($\mathbf{N} = 8 \cdot 30 = 240$) we get $c(\overrightarrow{z'}) = -0.05$ and $P(C \geq -0.05) = 0.7994$ which constitutes the effect as statistically not significant. Therefore, the null hypothesis H_2 is not rejected and it seems that observed human judgement of interest does not correlate with performance during play.

However, before abandoning the hypothesis of the performance impact on human judgement totally, we attempt to draw the relation between the two from another perspective. Figure 7.1 illustrates a scatter plot of the correlation coefficients between the

performance and the subject's judgement of interest against the correlation coefficients between the interest metric and the subject's judgement of interest (see Section 7.3.1) for each subject. In addition, the line $f(x) = -0.5864x$ (see (7.8)) of the statistical correlation between the two samples of data is plotted, where

$$f(x) = cor(c(\overrightarrow{z}), c(\overrightarrow{z'})) \cdot x \tag{7.8}$$

and

$$cor(c(\overrightarrow{z}), c(\overrightarrow{z'})) = \frac{cov(c(\overrightarrow{z}), c(\overrightarrow{z'}))}{(\sigma_{c(\overrightarrow{z})}\sigma_{c(\overrightarrow{z'})})} \tag{7.9}$$

If we examine Figure 7.1 in detail as well as answers in question C.2 (see Table E.2 in Appendix E), there seems to be a classification of the subjects into three groups. These are

- Subjects that judge interest according to their performance ($c(z') \geq 0.5$), size: 6 out of 30 subjects. As far as their agreement with the interest metric is concerned, it is not clear and their observed judgement portrays a rather random behavior ($0.0 \leq c(z) \leq 0.25$). Answers obtained from question C.2 are very explicit. Scoring performance and randomness are the major criteria in selecting the most interesting set between two. Subjects of this category are denoted by the numbers 5, 9, 10, 13, 14 and 22 in Table E.2.

- Subjects that do not judge interest according to their performance ($c(z') \leq 0.0$) and whose interest judgement correlates with the interest metric ($c(z) \geq 0.5$), size: 12 out of 30 subjects. This group's answers to C.2 are mainly focused on the opponent's contribution to the player's satisfaction. Subjects of this category are denoted by the numbers 2, 4, 7, 8, 11, 16, 18 and 26–30 in Table E.2.

- Subjects that do not judge interest according to their performance ($-0.5 < c(z') < 0.5$) and whose interest judgement does not correlate with the interest metric ($c(z) < 0.5$), size: 12 out of 30 subjects. Subjects of this group seem to concentrate on the opponent behavior as well as on a variety of Pac-Man aspects different or implicity syngeneic to the *Ghosts*' behavior, as acquired from answers on the C.2 question. These aspects include performance, game control ability, graphics, difficulty and duration of game. Subjects of this category are denoted by the numbers 1, 3, 6, 12, 15, 17, 19–21 and 23–25 in Table E.2.

The computed statistical correlation value and Figure 7.1 provide evidence that human judgement of interest, that agrees with the interest metric, is not correlated with the human judgement of interest based on performance. In other words, it seems that subjects agreeing with the interest metric do not judge interest by their performance. Or else, subjects disagreeing with the interest metric seem to judge interest by their score and/or other criteria such as game controls and graphics. Finally, as seen from Table E.2 subjects appear to agree on the conceptual definition of the three interest criteria for predator/prey games which are namely challenge, diversity in *Ghosts*' behavior and spatial diversity. Moreover, the majority of the subjects also identifies these as crucial factors for a good Pac-Man game before playing it — see Table E.1 in Appendix E.

7.3.5.1 Opponent 1

For the performance factor we also examine the case of $I_1 > I_2$. The case of $I_1 = I_2$ is not investigated since the 1–2 pair is not taken into consideration and z' values cannot be computed for subjects that played that particular pair of sets. Thus, following the same procedure as in section 7.3.5 and assuming that $I_1 > I_2$, we get $c(\overrightarrow{z'}) = -0.0667$ and $P(C \geq -0.0667) = 0.8639$ which constitute performance as a non significant factor for the human judgement of interest. In addition, experiments on this assumption reveal a slightly higher statistical correlation value $cor(c(\overrightarrow{z}), c(\overrightarrow{z'})) = -0.5341$, but conceptually the same effects and subject classification groups as the above-mentioned.

7.4 Conclusions

In this chapter we managed to confirm our hypothesis that the interest value computed by (2.5) is consistent with the judgement of human players by testing the game against human subjects. In fact, human player's notion of interest of the Pac-Man game seems to correlate highly with the captured interest value. In addition, it is revealed that both the subject type (i.e. experience with the game) and the order of playing the game do not affect their judgement. Moreover, given each subject's game score, it was demonstrated that humans agreeing with the interest metric do not judge interest by their performance; humans disagreeing with the interest metric judge interest by their score or based on other personal criteria like game control and graphics.

As far as on-line learning against human players is concerned, results show that more on-line learning games are required for the interest value to change significantly and for humans to notice some sort of change in the interest of the game. More computing power, through on-line gaming servers, may prove an efficient solution to this problem. This way, thousands of mutants could be evaluated in parallel over longer periods — which would provide better behavior estimates — and moreover the frequency of evolutionary iterations could be increased. Using this approach we accelerate the learning where appropriate and minimize the probability of unwanted, unrealistic, non-intelligent generated behaviors due to mutation. It is a fact that a single unrealistic emerged AI behavior is sufficient to impair the 'intelligent' image any adaptive approach is attempting to present and furthermore to diminish the satisfaction of the player (Champandard, 2004; Funge, 2004).

The main assumption about the Pac-Man game when the interest metric was formulated is that players overall have a basic level of gaming skills for the particular game. In that sense, the computer-guided players used are models of some well behaved, average skill players based on similar motion patterns that do not leave much space for significant differences in their performance and the OLL experiment best generated interest values. Humans players that tested this game cross-validate this assumption since their generated interest values against the same opponent were not significantly different from each other.

The next chapter presents a more sophisticated on-line learning mechanism developed for overcoming the long convergence time of OLL. This mechanism embeds a player modeling technique for faster adaptation while learning in real-time.

Chapter 8

The Player

"*I don't have any problem with any of the ghosts. Remember, I'm perfect.*"

Billy Mitchell, Pac-Man world champion.

The work presented in this thesis is mainly concentrated on the opponents' contribution to generating entertaining predator/prey computer games. However, the player through his/her playing skills and the real-time interaction with the opponent may also play an important role in obtaining more enjoyable games. For instance, we saw how diverse playing styles may cause big variations in entertainment in the Dead End game. Thus, beyond the opponent, the first additional entertainment factor to explore is the player per se.

The first step in the player aspect of entertainment is presented comprehensively in this chapter, where in Section 8.1 the player's impact on his/her entertainment is investigated through a model of his/her real-time actions[1]. In particular, players are classified according to their playing style and a form of linkage between the player type and the on-line learning mechanism is activated. We show that such a linkage leads to generation of more entertaining games for the player in less time. Moreover, the proposed approach demonstrates high adaptability into dynamical playing strategies as well as reliability and justifiability to the game user.

We believe that this human-game interaction should be expanded through innovative means so that it can cover important features of human players. Future steps on the

[1]This work was published in collaboration with Dr. Maragoudakis (Yannakakis and Maragoudakis, 2005). His contributions include the methodology and practice of the Bayesian Network training.

player aspect which include his/her emotional and cognitive perspective are presented in Section 8.2. Given that game graphics and sound systems have reached their limitations and nowadays slow steps are made towards their improvement (Champandard, 2004) — which can have an impact on player's entertainment — the exploitation of a potential relation between the player's style, emotional state and the game's generated entertainment will give further insights for the development of more advanced and enjoyable computer games.

8.1 Player Modeling

In the work presented in this section, we attempt to study the player's contribution to the emergence of entertaining games. We do that by investigating a Player Modeling (PM) mechanism's impact on the game's interest when it is combined with the proposed on-line learning procedure. More specifically, we use Bayesian Networks (BN), trained on computer-guided player data, as a tool for inferring appropriate parameter values for the chosen on-line learning mechanism. On-line learning is based on the idea of opponents that learn while they are playing against the player which, as already seen, leads to games of high interest. However, the parameters e_v and p_m strongly influence the performance of the on-line learning mechanism. Naive selection of these values may result in disruptive phenomena on the opponents' behavior through the mutation operator. Section 8.1.4 presents a Bayesian Network based mechanism designed to lead to more careful OLL parameter value selection and furthermore to an increasingly interesting game. It also stresses the correlation of the player's actions with the e_v and p_m value selection.

For the experiments presented here the Pac-Man game is used as a test-bed. In particular, all methods are tested in the Normal stage (see Figure 6.1(c)) as being the stage of medium complexity among the stages used. That is because, at this point in the thesis, we are primarily interested on player modeling's impact rather than the game's complexity (for experiments on the OLL method's effectiveness versus the Pac-Man game's complexity, see Chapter 6)

Results obtained show that PM positively affects the OLL mechanism to generate more entertaining games for the player. In addition, this PM-OLL combination, in comparison to OLL alone, demonstrates faster adaptation to challenging scenarios of frequent

changing playing strategies.

8.1.1 Player Modeling in Computer Games

Player modeling in computer games and its beneficial outcomes have recently attracted the interest of a small but growing community of researchers and game developers. Houlette's (2004) and Charles' and Black's (2004) work on dynamic player modeling and its adaptive abilities in video games constitute representative examples of the field. According to Houlette (2004), the primary reason why player modeling is necessary in computer games is in order to recognize the type of player and allow the game to adapt to the needs of the player. Many researchers have recently applied such probabilistic network techniques for player modeling on card (Korb *et al.*, 1999) or board games ((Vomlel, 2004) among others) in order to obtain adaptive opponent behaviors — see also (Hy *et al.*, 2004) for a bayesian programming application for more efficient first-person shooter (FPS) characters.

8.1.2 Bayesian Networks

A BN consists of a qualitative and quantitative portion, namely its structure and its conditional probability distributions respectively. Given a set of attributes $A = \{A_1, \ldots, A_k\}$, where each variable A_i could take values from a finite set, a Bayesian Network describes the probability distribution over this set of variables. We use capital letters as X, Y to denote variables and lower case as x, y to denote values taken by these variables. Formally, a BN is an annotated Directed Acyclic Graph (DAG) that encodes a joint probability distribution. We denote a network B as a pair $B =< S, P >$ (Pearl, 1988) where S is a DAG whose nodes correspond to the attributes of A. P refers to the set of probability distributions that quantifies the network. S embeds the following conditional independence assumption: Each variable A_i is independent of its non-descendants given its parent nodes. P includes information about the probability distribution of a value a_i of variable A_i, given the values of its immediate predecessors in the graph, which are also called "parents". This probability distribution is stored in a table, called the conditional probability table. The unique joint probability distribution over A that a network B describes can be computed using 8.1.

$$p_B(A_1,\ldots,A_k) = \prod_{i=1}^{k} p\left(A_i|parents(A_i)\right) \tag{8.1}$$

8.1.2.1 BN for Classification

Classification is a fundamental concept in the fields of data mining and pattern recognition that requires the construction of a function that assigns a target or class label to a given example, described by a set of attributes. This function is referred to as a "classifier". Given a set of pre-classified instances, numerous machine learning algorithms such as neural networks, decision trees, rules and graphical models, attempt to induce a classifier, able to generalize over the training data.

While Bayesian graphical models were known for being a powerful mechanism for knowledge representation and reasoning under conditions of uncertainty, it was only after the introduction of the so-called Naïve Bayesian classifier (Duda and Hart, 1973) that they were regarded as classifiers, with a prediction performance similar to state-of-the-art classifiers. The Naïve Bayesian classifier performs inference by applying Bayes rule to compute the posterior probability of a class C, given a particular vector of input variables A_i. It then outputs the class whose posterior probability is the highest. Regarding its computational cost, inference in Naïve Bayes is feasible, due to two assumptions, yet often unrealistic for real world applications:

- All the attributes A_i are conditionally independent of each other, given the classification variable.
- All other attributes are directly dependent on the class variable.

Despite the fact that Naïve Bayes performs well, it is obviously counterintuitive to ignore the correlation of the variables in some domains.

8.1.2.2 Learning General Bayesian Networks from Data

There are two practices for determining the structure of a Bayesian Network: Either manually, by a human domain expert who should provide the interconnection of the variables, or having the structure determined automatically by learning from a set of training examples. Regarding the learning of the conditional probability table of a network, the same principle applies. The parameters of the table could either be provided

manually by an expert or automatically through a learning procedure. The task of manually supplying the parameters is a laborious one. Besides, in some applications it is simply infeasible for a human expert to know *a priori* both the structure and the conditional probability distributions.

8.1.3 Bayesian Networks for Player Modeling

Bayesian Networks (Pearl, 1988) provide a comprehensive means for effective representation of independent assumptions. Moreover, they can provide a mechanism for effective inference under conditions of uncertainty. More specifically, they have been extensively used in a variety of fields such as pattern recognition (Mitchell, 1997), natural language processing (Maragoudakis *et al.*, 2004), decision support (Horvitz and Barry, 1995), etc. In the field of player modeling, BN can cope with the significant issue of uncertainty on the model of the player, allowing for inference on the class variable given a subset of the input features, rather than a complete representation of them. Such a feature is significant in domains where modeling of a human is required. The ability of BN to adapt to new domain characteristics is also beneficial for PM applications since the belief in a characteristic is prone to change from time to time, depending on the condition of the game.

For PM in our test-bed we used the Bayesian Network Augmented Naïve Bayes (BAN), by Cheng and Greiner (2001). In a BAN (see Figure 8.1), all attribute nodes are children of the class node (C) and they form a general BN. This approach combines the ability of general BN to encode the interdependencies of the input variables with the classification bias posed by the BAN structure, a factor that ensures enhanced performance on predicting the class variable over the general BN approach. Section 8.1.4 presents the application of PM combined with the OLL mechanism.

8.1.4 PM-OLL Mechanism

As previously mentioned, the primary goal of this work is to investigate whether player modeling can contribute to the satisfaction of the player. Towards this aim we combine PM, by the use of BN, with the OLL algorithm to form the PM-OLL mechanism presented here. The two mechanisms' interaction flows through the OLL parameters which are set by inferences from the PM mechanism (see Figure 8.2).

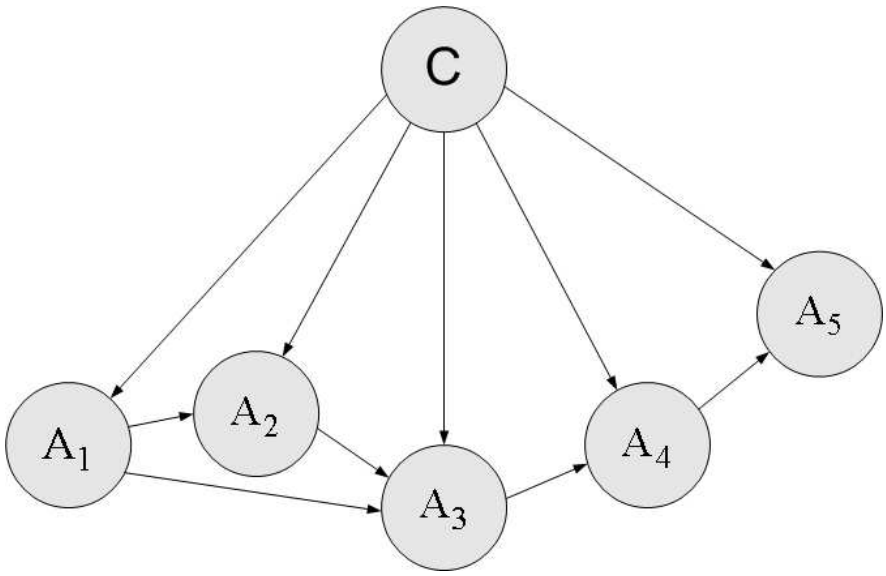


Figure 8.1: The BAN structure.

For the present work, in order to construct a model of the player, we have considered the following features obtained from an on-line learning play of 10 games:

1. Score (i.e. total number of pellets eaten).
2. Time played in simulation steps.
3. Grid-cell visits entropy of the player — this metric corresponds to the player's spatial diversity.
4. Initial interest of the game.
5. Relative interest difference after 10 games are played.
6. Evaluation period e_p in simulation steps.
7. Probability of mutation p_m.

Our objective, given that we desire maximum interest augmentation in the game, is to find the optimal values for the features p_m and e_p, given the player input variables (i.e. score, time played, entropy, initial interest). In other words, by using PM-OLL we attempt to investigate the correlation between a player's actions and appropriate values for the OLL parameters able to increase the interest of the game.

The previously described BN (see Section 8.1.3), which embodies the PM mechanism, is trained off-line on feature instances. The total number of training data is 2200, ob-

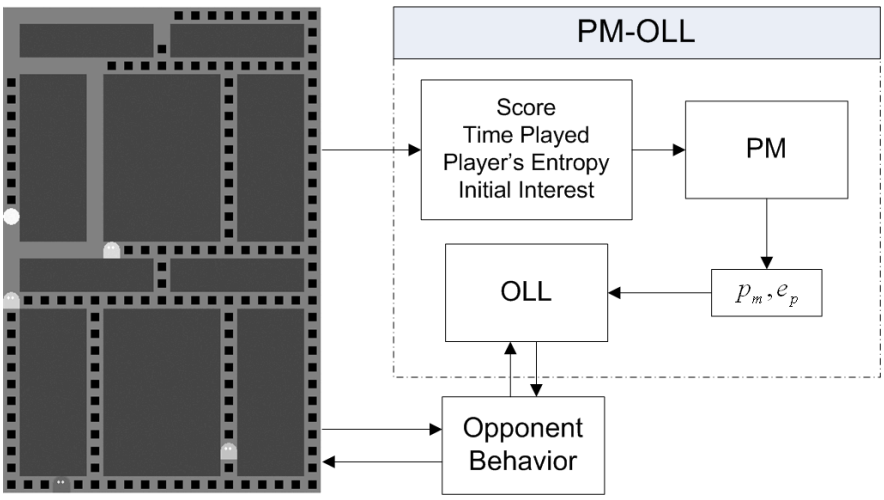


Figure 8.2: The PM-OLL mechanism.

tained by multiple on-line learning simulation runs of variant opponents against all three different hand-crafted *PacMan* types (see Chapter 6 for more details) within a fixed set of e_v and p_m values. These sets are empirically selected to be $p_m \in \{0.005, 0.01, 0.02, 0.03, 0.05, 0.1, 0.2, 0.5, 1.0\}$ and $e_p \in \{2, 5, 10, 25, 50\}$ since we believe that these correspond to representative and reasonable values of the two OLL parameters' intervals.

In a more mathematical manner, the aim is to maximize the probabilities

$$P(e_p|Score = value1, \ldots, VARN-1 = valueN, Interest\ Change = max) \quad (8.2)$$

and

$$P(p_m|Score = value1, \ldots, VARN-1 = valueN, Interest\ Change = max) \quad (8.3)$$

8.1.5 Results

8.1.5.1 BN Training

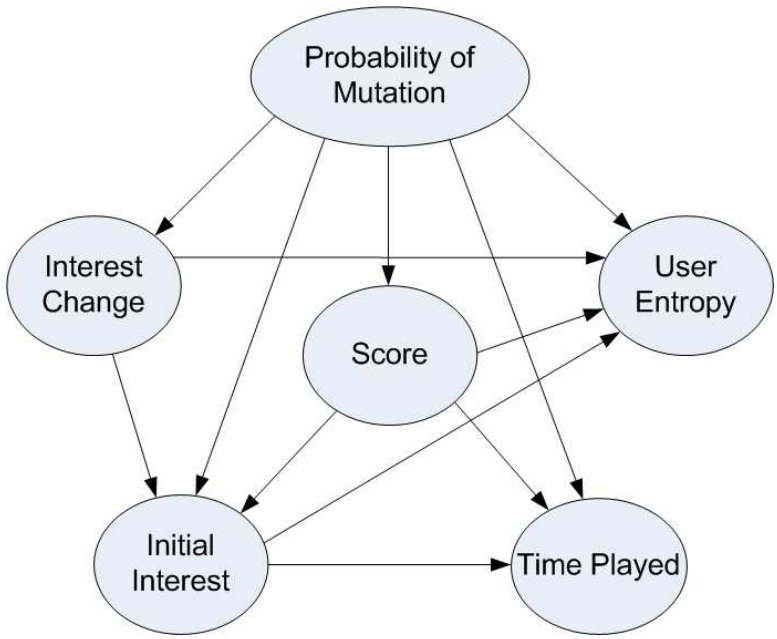
The problem of finding the most probable network structure from data is known to be NP-hard (Mitchell, 1997) meaning that there are $2^{\frac{n(n-1)}{2}}$ possible networks that could describe n different attributes. For that reason, we have utilized the *Bayesian scoring*

function (Heckerman *et al.*, 1995) which provides a metric of relation among two candidate networks. Regarding the search among the plethora of candidate networks, we used the following approach in order to achieve a computationally effective strategy: Initially, the most probable forest-structured network is constructed. A greedy search is performed by adding, deleting or reversing the arcs randomly. If a change results in a more probable network it is accepted, otherwise cancelled. Throughout this process, a repository of networks with high probability is maintained. When the search reaches a local maximum, a network is randomly selected from the repository and the search process is activated again. The network complexity is controlled during the search, so that a limited number of arcs is allowed in the beginning and, as the process progresses, more and more arcs are approved. Upon completion of the BN structure learning mechanism (see Figure 8.3), the Expectation-Maximization (EM) algorithm (Dempster *et al.*, 1977) is used in order to estimate the parameters of the conditional probability table.

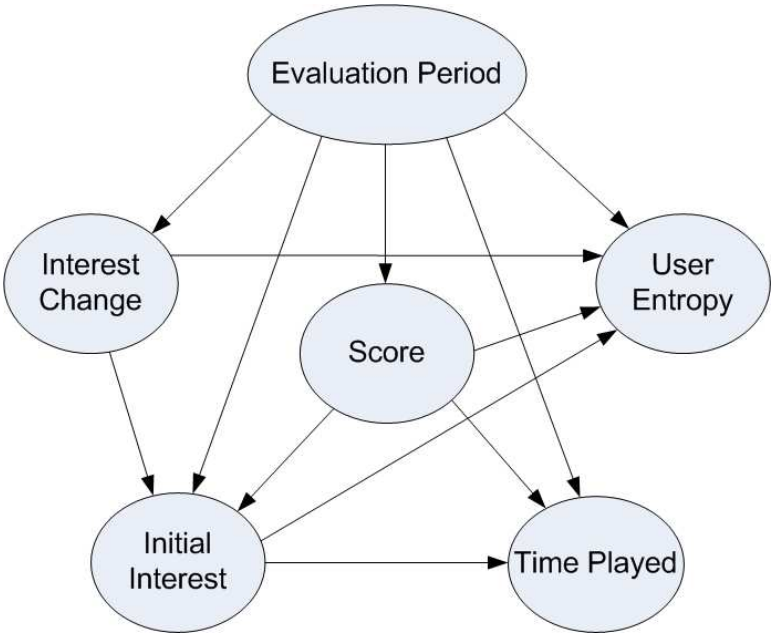
8.1.5.2 Adaptability Tests

As already presented in Chapter 6, in order to effectively test a learning mechanism's ability to adapt to a changing environment (i.e. change of player strategy), the following experiment is proposed. Beginning from the five initial behaviors of significantly different interest values we used in Chapter 7 (see Table 7.1) we apply the examined mechanism against a specific *PacMan* type. During the on-line process we keep changing the type of player every 20 games played. The process stops after 60 games when all three types of player have played the game. Since there are three types of fixed strategy players, the total number of different such experiments is 30 (6 different player type sequences times 5 different initial behaviors).

An alternative fashion of investigating adaptability is to let each group of *Ghosts* play against randomly chosen opponents for a single longer period (e.g. 200 games). The random selection, which occurs every 20 games, is made via a uniform distribution that assigns equal probabilities to each *PacMan* type. Both random and fixed *PacMan* selection adaptability tests (scenarios) illustrate the overall picture of the mechanism's behavior respectively against a random and any fixed sequence of the computer-guided *PacMan* types.



(a) p_m



(b) e_p

Figure 8.3: The BAN trained structures for p_m and e_p.

8.1.5.3 OLL Parameter Selection

In Chapter 6 and Chapter 7, we empirically chose the OLL parameters to be $p_m = 0.02$ and $e_p = 25$. Experiments with these parameter values demonstrated high robustness

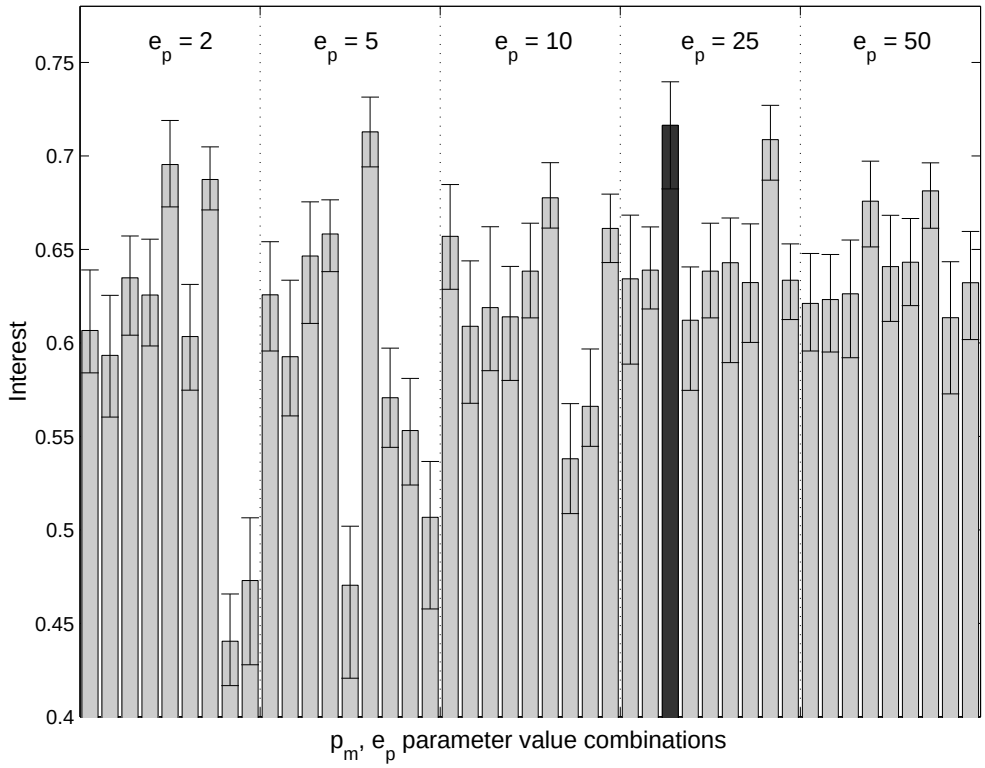


Figure 8.4: Interest values of all different pairs of (p_m, e_p) parameters. The dark gray bar indicates the $(0.02, 25)$ pair of values.

and adaptability of the OLL mechanism. In the work presented here we will attempt to conduct a sensitivity analysis on these parameters to test our parameter selection. By picking all pairs (p_m, e_p) from the fixed sets of parameter values and applying the adaptability test of a single fixed *PacMan* selection scenario (see Section 8.1.5.2) for the OLL, for each one of the parameter pairs, we obtain a significant number of parameter value pairs (i.e. 45) to study. For each pair of parameter values, the average interest and confidence interval values achieved over the 60 games played in total are presented in Figure 8.4.

As seen from Figure 8.4, the empirically selected combination demonstrates the highest interest value ($I = 0.7168$) amongst all 45 tested. In addition, it presents the lowest variance ($2 \cdot 10^{-4}$) of interest among the other 10 parameter pair values that generated non-significantly different interest values during the 60 game period of testing. Given these statistics, it appears that the $(0.02, 25)$ values constitute the most appropriate pair of fixed OLL parameters and, therefore, are selected for all experiments presented here.

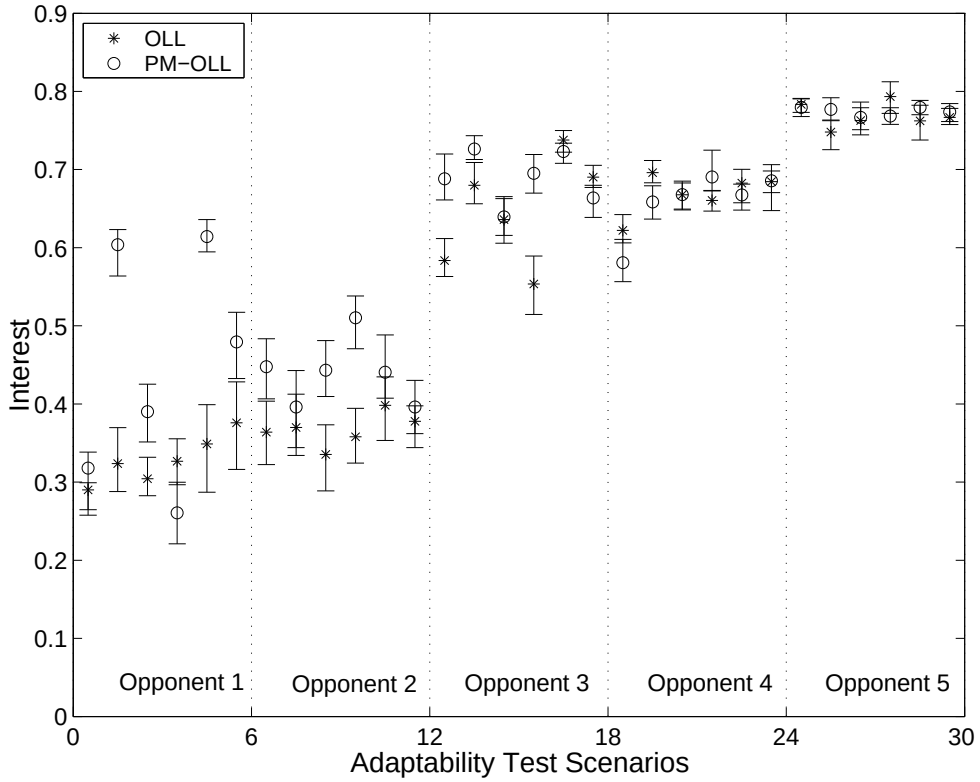


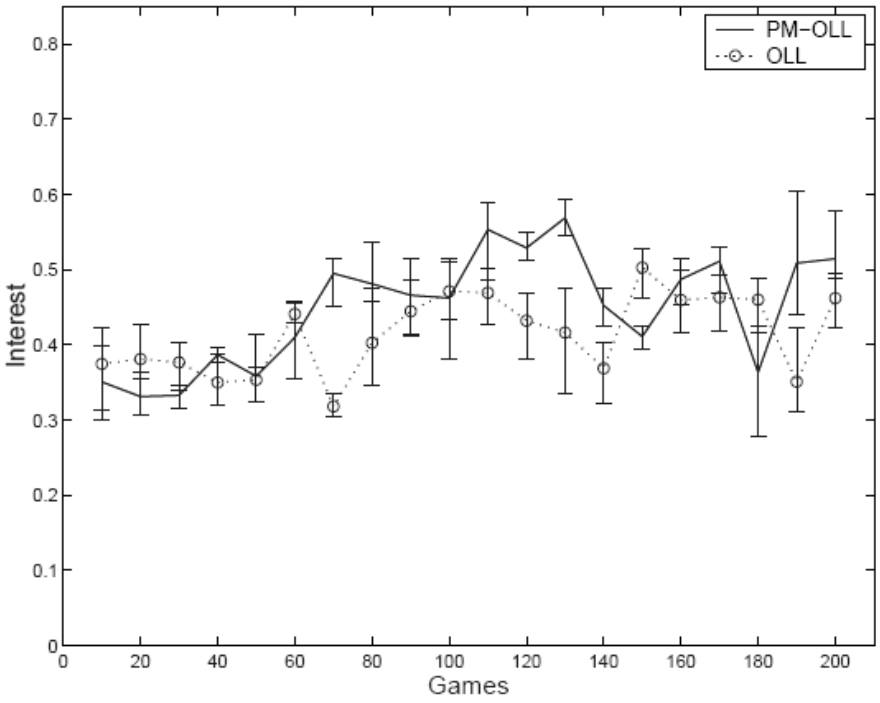
Figure 8.5: Adaptability tests with fixed *PacMan* selection: Average interest values and interest intervals generated by OLL and PM-OLL for all 30 different game scenarios.

8.1.5.4 Comparative Study

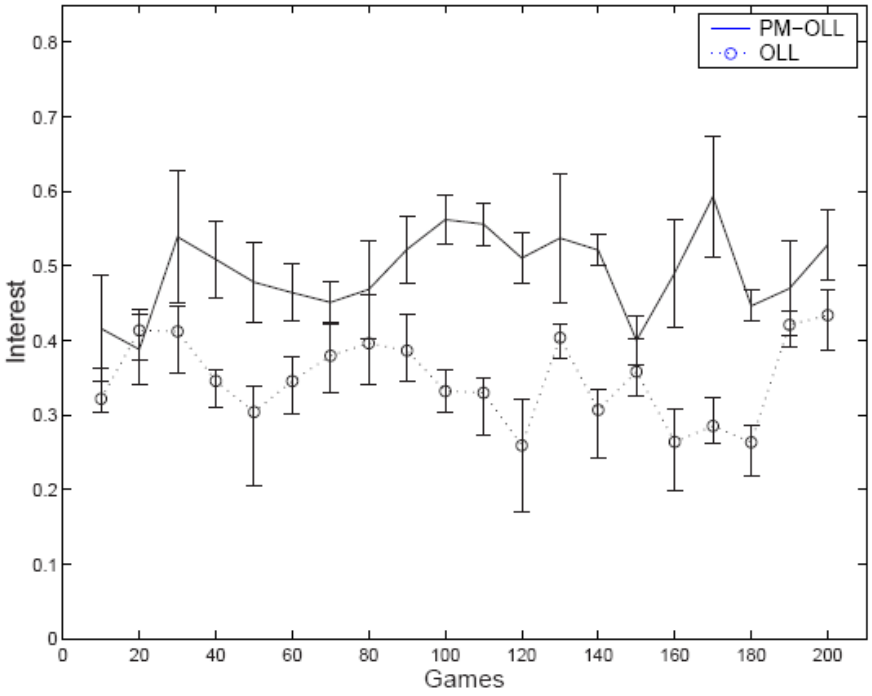
To test the PM impact on the OLL mechanism's ability to generate games of high interest we first apply the adaptability test of fixed *PacMan* selection (see Section 8.1.5.2) for the OLL alone (fixed parameter values — $p_m = 0.02, e_p = 25$) and for the PM-OLL approach. For the latter, player modeling occurs for every 10 games played inferring values for p_m and e_p.

Results obtained (see Figure 8.5) demonstrate that the PM-OLL mechanism is able to generate more interesting games than the OLL mechanism in 23 out of 30 playing scenarios examined; in 12 of these cases the difference is statistically significant. Given this comparative study, it appears that the player modeling impact on the generation of interesting games is positive and that the PM-OLL mechanism is independent of the sequence of the changing *PacMan* type and the initial opponent.

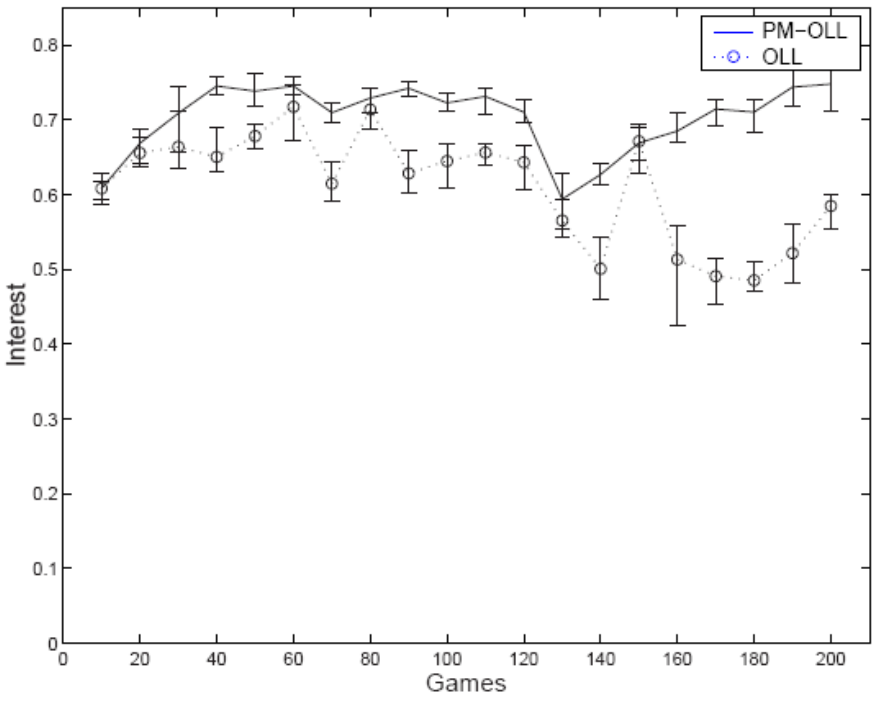
Alternatively, the adaptability test with random *PacMan* selection is applied for 200 games beginning each of the five different *Ghost* behaviors. We believe that 200



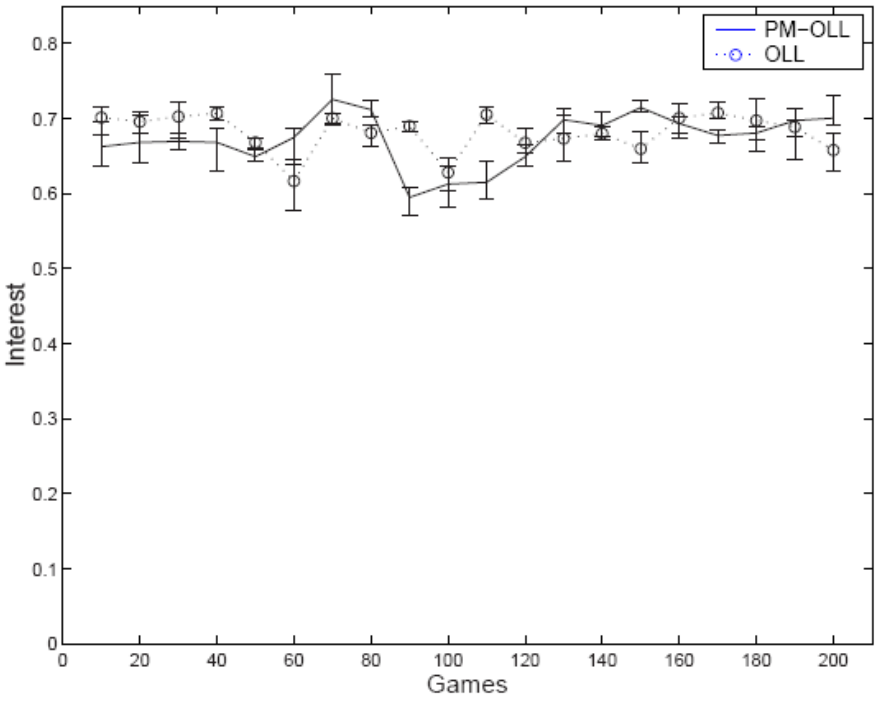
(a) Opponent 1



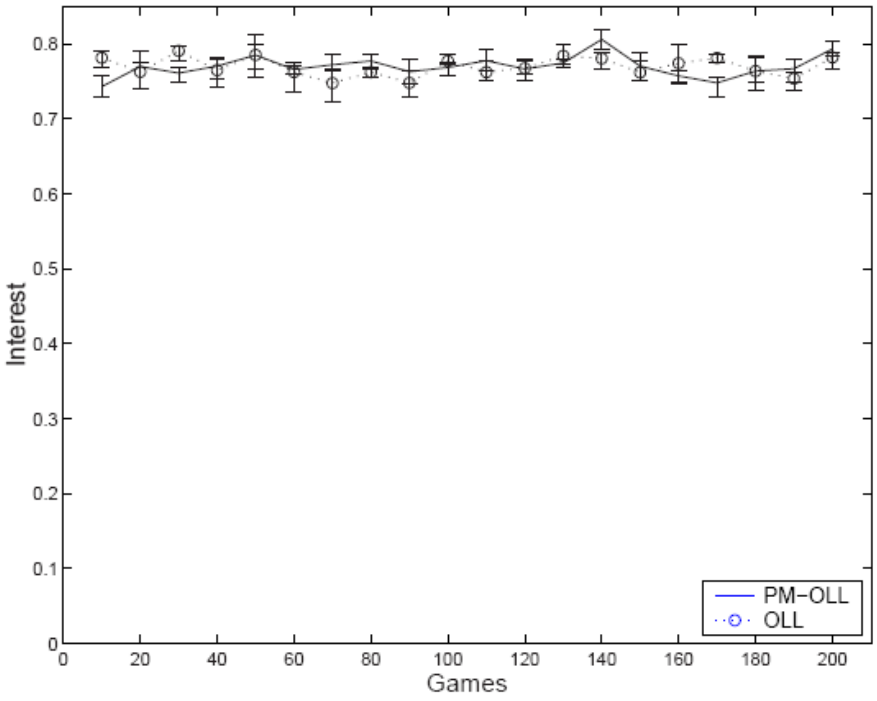
(b) Opponent 2



(c) Opponent 3



(d) Opponent 4



(e) Opponent 5

Figure 8.6: Adaptability tests with random *PacMan* selection.

games are adequate to display the non-deterministic effect of the *PacMan* selection in the mechanisms examined, since when fixed *PacMan* selection occurs on-line learning adaptive behaviors are revealed within 60 games. Figure 8.6 illustrates the evolution of interest throughout the games for both OLL and PM-OLL mechanisms. Table 8.1 shows the average interest value generated from these mechanisms during the experiments. As in the experiments with fixed *PacMan* selection, the PM-OLL mechanism demonstrates an adaptive behavior which generates interesting games faster and more efficiently than OLL alone. In two scenarios (i.e. initial opponent 2 and 3) the generated average interest value of PM-OLL is significantly higher than the respective OLL value. For the other three scenarios the difference in insignificant (Table 8.1). Overall, the comparative study results provide evidence for the PM-OLL mechanism's ability to quickly adapt to new playing strategies as well as its efficiency in producing games of high interest against human players, faster than OLL.

	PM-OLL			OLL		
Initial Opponent	$E\{I_u\}$	$E\{I\}$	$E\{I_l\}$	$E\{I_u\}$	$E\{I\}$	$E\{I_l\}$
1	0.491	0.473	0.414	0.454	0.436	0.378
2	0.531	0.512	0.442	0.385	0.367	0.301
3	0.721	0.712	0.681	0.639	0.625	0.578
4	0.691	0.682	0.652	0.697	0.689	0.662
5	0.782	0.776	0.756	0.783	0.776	0.752

Table 8.1: Adaptability tests with random *PacMan* selection: Average interest $E\{I\}$ and confidence interval $(E\{I_u\}, E\{I_l\})$ values.

8.1.5.5 Randomness Testing

In this section we will test the hypothesis that random selection of the set of p_m and e_p values, instead of Bayesian Network produced values, has a better impact on the generated interest value of the game. In order to assess the truth of this hypothesis we apply the adaptability test for PM-OLL where p_m and e_p values are picked randomly and not through the BN. The test is conducted for all thirty adaptability test scenarios with fixed *PacMan* sequence selection (see Section 8.1.5.4). The outcome of this experiment is illustrated in Figure 8.7 where scenarios are ranked by their generated average interest value when random parameter selection is made.

It turns out that our hypothesis is not valid since, in 27 out of 30 cases examined, the random fashion of selecting values for p_m and e_p generates lower average interest value than the interest value generated by PM-OLL. More specifically, in 16 cases this interest value difference is statistically significant (see Figure 8.7). These results imply that appropriate selection of OLL parameter values correlates to the improvement of the player's satisfaction, and moreover, the proposed BN approach of selecting OLL parameter values does indeed have a positive impact on the game's emergent interest value.

8.1.6 Conclusions

Successful applications of the on-line learning approach (see Chapter 5 and Chapter 6) have already shown the mechanisms' robustness in generating predator/prey computer games of high interest and fast adaptability to changing playing strategy situations. As seen in Chapter 7 the suggested interest metric for a predator/prey game correlates

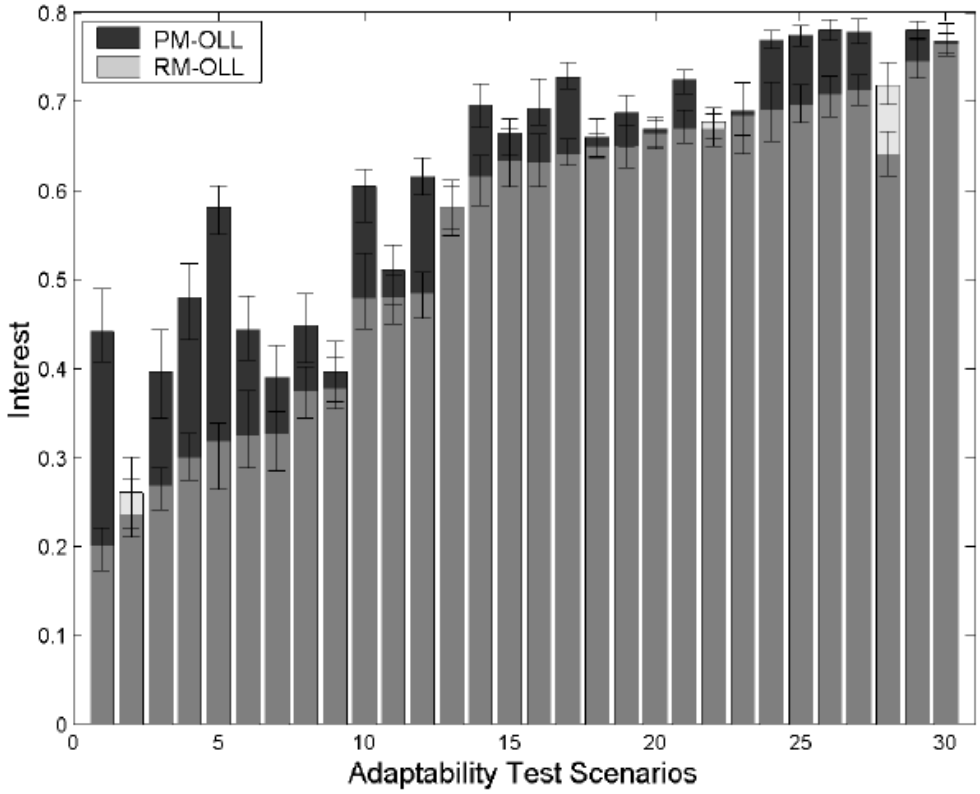


Figure 8.7: Randomness hypothesis testing: Randomly selected parameter values (RM-OLL) against parameter values proposed by the BN (PM-OLL).

with human judgement of interest, that is, human players' notions of interest of the Pac-Man game seem highly correlated with the proposed measure of interest.

In this chapter the player's, in addition to the opponent's, impact on the game's interestingness was investigated. In particular, we demonstrated a PM mechanism's positive impact on the generation of more interesting games by using the Pac-Man Normal stage (see Figure 6.1(c)) test-bed. Moreover, the proposed PM-OLL mechanism shows game reliability since it demonstrates adaptive behaviors in the scale of tens of games played and it is computationally inexpensive (1–3 seconds of CPU time for the BN to infer OLL parameter values; few milliseconds for the OLL to evaluate the *Ghost* population on-line).

8.2 The Road Ahead

Given the current state-of-the-art in AI in games, it is unclear which features of any game contribute to the enjoyment of its players, and thus it is also doubtful how to generate enjoyable games. Research endeavors aiming to do this without clear understanding of what factors yield enjoyable gaming experience will inevitably be unsuccessful or will succeed at best by accident. Unfortunately, most commercial and some academic research in this area is deficient in this area — as demonstrated by its lack of success in providing the more interesting games humans evidently seek.

In order to bridge the current gap between human designation of entertainment and interest generated by computer games and to find efficient and robust paths in obtaining appealing games, there is a need for an intensive and interdisciplinary research within the areas of AI, human-computer interaction and emotional and cognitive psychology. The proposed future research aims at exploring the novel directions opened by this thesis on introducing entertainment measurements and adaptive learning tools for generating interesting computer games. The long-term target of this work is to reveal the direct correlation between the player's perceived entertainment (I), his/her playing strategy (style) (U) and his/her emotional state (E) — see Figure 8.8. Such a perspective will give insights into how a game should adapt to and interact with humans, given their emotional state and playing skills, in order to generate high entertainment. Towards this purpose, an innovative computer game will be developed, based on an interactive system that will allow one to study the ongoing processes of situated game state, the user's playing style and emotional flow. Such a system will embed automatic questionnaires and dialogues as well as video and sound recording of the user.

As an outline, the objectives arising towards the future work's key target are as follows (see Figure 8.8):

- Test existing state-of-the-art methodology, primarily based on the ideas in this thesis, in more complex cooperative games. Popularity, open source platform and on-line gaming potential are the basic game selection criteria, which leave space for FPS and/or real-time strategy (RTS) massively multi-player on-line games (MMOG).

- Construct an on-line web server that will host all AI processes (e.g. on-line adaptive learning) for computational effort reasons. The server will monitor game

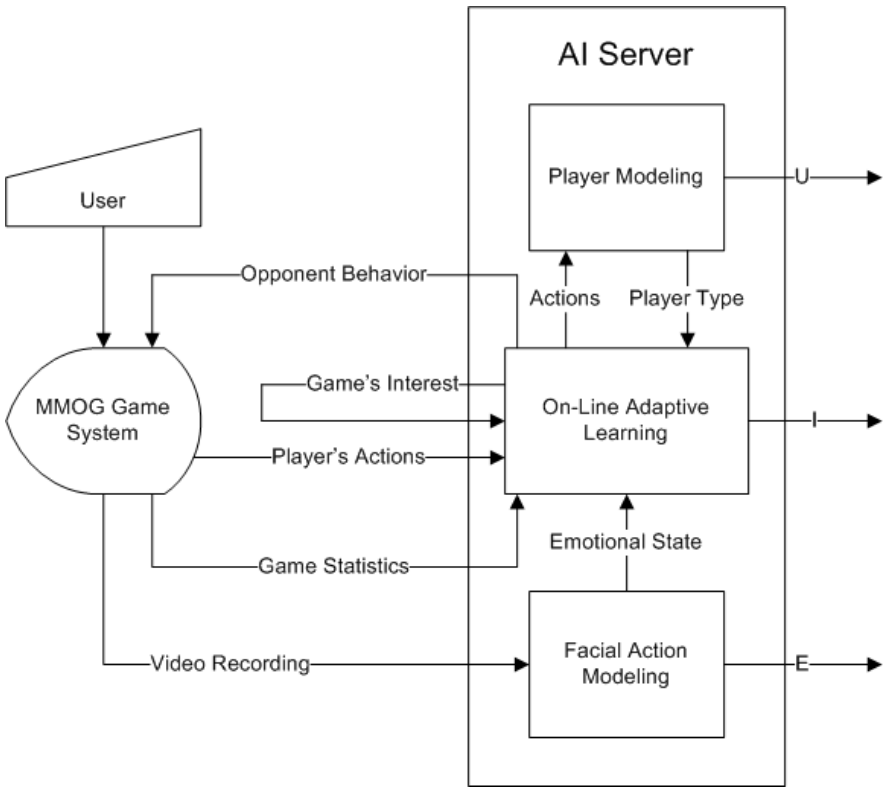


Figure 8.8: The future work scheme.

features (e.g. player's actions), reinforce the AI with its real-time generated interest I and adjust the opponent behavior back to the game.

- Record players' actions real-time in order to dynamically model the player and classify him/her into a player type U, which will determine features of the AI adaptive processes (e.g. on-line learning mechanism).

- Record players' behavior real-time by audiovisual means in order to model their dynamical emotional flow E through facial coding procedures. The player's emotional state will also determine features of the adaptive on-line learning mechanism. For modeling the player's emotions in real-time, we gain inspiration primarily from the work of Kaiser et al. (1998). They attempted to extract basic emotions in real-time — according to the Facial Coding Action System (Eckman, 1979) — by recording facial expressions of humans playing a predator/prey computer game similar to Pac-Man (Kaiser and Wehrle, 1996).

The potential of this future work lies in its innovative endeavor to bring emotional psychology, human-machine interaction and AI advanced techniques to meet upon a

computer game platform. As soon as experiments with statistically significant numbers of human subjects are held, this work's outcome will provide important insights to spot the features of computer games — that map to specific emotions — that make them appealing to most humans. Moreover, the playing strategy and emotional features that generate entertainment in games will be exposed which will open new horizons for the game AI research community.

8.3 Summary

This chapter, unlike the preceding technical chapters of this thesis, was devoted in exploring the player's contribution to its entertainment. Initial steps towards this direction reveal that by modeling the user's actions which reinforce the on-line learning procedures, one might come up with more interesting games. Furthermore, this entertainment enhancement takes place much faster than when OLL is used alone even if the player's strategy changes randomly and frequently (i.e. every 20 games).

The subsequent steps of this work lie in exploiting the current knowledge on how to generate interesting games by adjusting opponent behavior on-line to further examine the player's perspective on computer games. In particular, the proposed future research is framed by the player's both recorded real-time actions and emotional state which may determine features of the on-line adaptive learning mechanism.

In the following and last chapter of this thesis we will summarize the thesis' main achievements and contributions and discuss the proposed methods' current limitations. Moreover, potential solutions that might embrace these drawbacks are presented.

Chapter 9

Conclusions

This thesis is primarily based on two research questions: how to measure entertainment for the player in computer games and which are the AI mechanisms that can effectively generate it. The predator/prey game genre was used as the initial step towards answering both questions. In particular, two dissimilar predator/prey games were devised as test-beds for our investigations.

Given our observations and empirical studies on this genre of games, we defined criteria that contribute to the satisfaction for the player which map to characteristics of the opponent behavior. According to our hypothesis, the player-opponent interaction — rather than the audiovisual features, the context or the genre of the game — is the property that primarily contributes the majority of the quality features of entertainment in a computer game. Based on this fundamental assumption, we introduced a metric for measuring the real-time entertainment value of predator/prey games. This value is extracted from three entertainment criteria: appropriate level of challenge, opponents' behavior diversity and opponents' spatial diversity.

According to our second hypothesis, enhanced entertainment is present when opponents demonstrate cooperative behaviors. By following principles of the animat approach (Meyer and Guillot, 1994) for realistic implicit and partial sensing in simulated worlds, we exhibited cooperative effective behaviors through off-line learning procedures in the *FlatLand* prototype simulated world. The conclusions derived from this world are that cooperation is plausible given a limited sensing scenario and that unsupervised mutation-based learning algorithms are more robust and computationally preferred than supervised learning techniques. In addition, the ECWAS algorithm was

used in order to automatically generate successful controllers of minimal size for the same test-bed.

Our third hypothesis is that entertainment is generated when adaptive learning procedures occur in real-time. That allows for opponents to learn while playing against the player and adapt with regards to his/her strategy. When such a mechanism is built upon cooperative opponents of minimal-sized controllers, it is more likely that the game's interest value improves drastically. Hence, we introduced a neuro-evolution on-line learning mechanism in Chapter 4. This approach embeds a replacement method of the worst-fit individuals while playing the game. The mechanism demonstrated robustness in enhancing the game's interest and adaptability against unknown playing strategies for both test-bed games used (see Section 6.11.1 and Section 6.11.2 for a discussion on the dissimilarities of the games and the OLL variants used respectively).

As soon as there was reasonable evidence for the OLL mechanism's generality over predator/prey game variants, initial opponent behavior, player type and stage complexity and topology, we introduced a human survey (see Chapter 7) aiming at cross-validating the interest metric proposed. Results have shown that our interest metric hypothesis is valid since humans appear to agree with this notion of entertainment. In addition it was found that neither the order of play, nor the subject type affect the judgement of humans as far as their entertainment is concerned. Moreover, given each subject's score, it was demonstrated that humans disagreeing with the interest metric judge interest by their performance or based on other individual criteria like game control, speed and graphics. Likewise, subjects agreeing with the I value have based their judgement on game features such as challenge, intelligent opponents, opponents' behavior diversity and spatial diversity. As far as the on-line learning experiment against human players is concerned, observations showed that fifty games are not adequate for the interest value to change significantly and for humans to notice a change in their perceived entertainment.

Since the use of PC games provides limited computational power, more sophisticated learning procedures were designed in order to accelerate learning in real-time. Hence, in Chapter 8 the player's contribution to the generation of entertainment was investigated. It was found that an enhanced OLL mechanism that embeds a successful player modeling tool appears to be more efficient and faster in generating the entertaining games that the player evidently seeks. In this chapter we also presented a scheme for obtaining computer games of richer interactivity and higher entertainment by focusing

on the real-time adjustment of the opponent's controller. The potential of the proposed scheme lies in its innovative endeavor to bring emotional psychology, human-machine interaction and advanced AI techniques to meet upon computer game platforms.

Given the experiments presented in this dissertation, the three aforementioned hypotheses are likely to be true: For the first hypothesis, human players confirmed that features of the opponent behavior constitute the main factors for distinguishing among games of different entertainment values and furthermore agreed on the interest metric proposed. For the second hypothesis, results demonstrated that cooperative action is present and maintained when the entertainment value of the game is high. For the third hypothesis, the robustness of the OLL mechanism in augmenting the game's interest and its adaptive features demonstrate that learning in real-time can enhance the entertainment value of a computer game.

9.1 Contributions

This section summarizes this thesis' main achievements and contributions to advance the state-of-the-art of AI in computer games and cooperative multi-agent systems. To a lesser degree, results of this thesis can be of use to the fields of emotional psychology, human-computer interaction and computer-enhanced education. More specifically, this dissertation has contributed the following:

- We introduced and established, through human player experiments, a generic measure for entertainment in predator/prey games.
- Based on our assumptions, we introduced and verified a generic methodology that increases the entertainment value of variants of predator/prey computer games. According to this, entertainment is enhanced when learning, which is built on cooperative multiple opponents of limited sensing and minimal controllers, acts in real-time. This methodology's features make it applicable to all multi-opponent games; careful design may guarantee its success in other genres of games (see Section 9.3).
- We introduced the ECWAS algorithm for automatically designing neural network controllers. ECWAS can be utilized in order to provide opponents with controllers of minimal size capable of achieving high performance in their tasks.

- The *FlatLand* world case study revealed that well-behaved cooperative opponents can emerge faster and more efficiently through unsupervised learning approaches rather than supervised learning techniques. These opponents can be used as initial good points for the generation of more entertainment games.
- The replacement and pre-evaluation methods in neuro-evolution on-line learning were successfully introduced in computer games through this thesis.
- Results demonstrated that cooperation is maintained among heterogeneous opponents during learning in real-time. This could explain the interesting games obtained through OLL: Cooperative opponents make games more appealing to the player.
- We introduced a method that uses player modeling techniques to adjust OLL parameters in real-time. This combination proved able to increase the opponent's adaptability and furthermore enhance the game's interest faster.

9.2 Limitations

The limitations of the proposed methodology, as appeared throughout the experiments of this thesis, are summarized in this section. They are classified into limitations considering learning in real-time in computer games and limitation concerning the interest metric proposed. Ideas for overcoming such drawbacks are discussed and provide the ground for future investigations.

9.2.1 Learning in Real-Time

The drawbacks that occur through the OLL application are outlined here. These follow the challenges that generally appear when designing learning mechanisms in real-time for computer games as presented in Chapter 1. As the reader will notice, the three limitations presented here are strongly correlated.

9.2.1.1 Computational Power

On-line learning suffers from convergence time. The mechanism would be able to adapt faster if more CPUs through a server were available. Against human play-

ers, OLL is not able to demonstrate more interesting opponents in 50 on-line learning games, since the maximum number of worst-fit replacements is only 600 for the Pac-Man game. We assume that by the use of more computational power the replacement frequency would increase and adaptability of OLL would be faster. In addition, several hundreds or thousands of mutants would be pre-evaluated off-line for longer periods of games contributing to the avoidance of poor behavior estimations (see also Section 9.2.1.2).

Appropriate OLL parameter adjustment via a probabilistic network pressed the learning procedure for a much faster adaptation and showed that player modeling might be an effective solution for the time convergence problem. The PM-OLL mechanism adapts even in environments of randomly generated playing strategies that change continuously (every 20 games).

9.2.1.2 Few Opponents

The interest value evolution over on-line learning games appears to be noisy and dependent on the number of opponents of the game. As seen from both games investigated, the I value is sensitive to opponent replacements since five is the maximum number of opponents in our experiments. Therefore, both highly interesting and boring opponent replacements may lead to drastic change of the entertainment value. Although the mechanism is designed (i.e. pre-evaluation mechanism, gaussian mutation, probability for the mutation operator) to prevent such occurrences for the changes to be smooth (see Section 9.2.1.3), undesired behaviors may still emerge.

This limitation defines one of the major challenges of learning on-line in computer games (see Chapter 1). Researchers and game developers have to build their learning mechanism on the assumption that the majority of games include a tiny number of opponents that the player can interact with in real-time. Additional challenge is generated since an on-line learning mechanism cannot act fast upon few player-opponent interactions. Case-injection algorithms (Miles and Louis, 2005) may be used to speed up the process. Cases of interesting NPCs may be kept in a 'hall-of-fame' list and injected the current opponent population as interesting opponents, when appropriate.

9.2.1.3 Time Versus Smoothness

The appropriate balance between computational time and smoothness of the opponent behavior is also a key question for the on-line learning procedure. No significant changes, unwanted or extremely unpredictable behaviors are desired in a game context because they are easily observable (given the few number of opponents) by the player. As previously mentioned, a single unintelligent opponent can destroy the 'intelligent' image of the game that OLL attempts to build (Champandard, 2004; Funge, 2004). For this issue a pre-evaluation method and a gaussian mutation operator were introduced in OLL for the game of Pac-Man. Even though smoothness of the interest value is achieved (see Section 9.2.1.2), the algorithm's convergence time is significantly increased given a single CPU (see Section 9.2.1.1).

9.2.2 Interest Metric

The I value proposed in Chapter 2 is a reliable estimate of player entertainment in real-time since it was approved by humans (see Chapter 7). This dissertation presented an innovative and efficient approach to model and quantify entertainment; however, without claiming of this approach being unique. We believe that other successful quantitative metrics for the appropriate level of challenge, the opponents' diversity and the opponents' spatial diversity may be designed and more qualitative criteria may be inserted in the interest formula.

The following interest metric limitations arise from the assumptions discussed in Chapter 2 and the I value's observed behavior.

9.2.2.1 Real-time Interest Estimation

As noted in Chapter 2, we require a number N of played games in order to effectively estimate the real-time interest value of the game. Even though this appears to follow human cognition in defining interestingness of a game, it constitutes a limitation of the method. A further investigation of the relationship between the I value and the N played games might reveal that fewer games are needed for an estimate that is still consistent with human notion of perceived entertainment.

9.2.2.2 Player Types

As far as the playing strategy is concerned, thirty human players have played Pac-Man and three hand-crafted players have been designed for each game. For the latter, independently of the playing strategy, the mechanism adapted to their playing style in order to boost the game's interest. More specifically, in the Pac-Man game, the best OLL generated interest values were not significantly different regardless of the player type. Likewise, the interest values generated by humans in this game against the same opponent, were not significantly different from each other. However, such a conclusion is not derived from Dead End, since the EA *Cat* tends to generate significantly more interesting games than the RM *Cat* and the PFB *Cat*.

Differences among the fixed strategy players used in Dead End are more clear varying from 'blind', as far as the opponents are concerned, random behaving players (i.e. RM *Cat*) to players that concentrate only to the Exit (i.e. EA *Cat*). These players break our assumption of average playing skills and therefore produce significant differences in their generated interest values. However, we did not face such poor playing strategies in our human players sample. Note that, some of the subjects have played Pac-Man only once or twice before the experiment.

9.3 Extensibility

Experiments over variants of predator/prey games of different complexity and their features have demonstrated the methodology's robustness throughout this dissertation. Here, we discuss the potential of the methodology in other genres of multi-opponent games where the complication of the opponents' tasks may differ. More specifically, we analyze the extensibility of the interest metric proposed, the on-line evolutionary learning mechanism and the neuro-controller used.

9.3.1 Interest Metric

As already mentioned in Chapter 2, the criteria of challenge and behavior's diversity may be effectively applied for measuring the real-time entertainment value of any genre of games. Spatial diversity may in a sense also contribute to the interest value of specific genres (e.g. team-sport, RTS and FPS games). As long as game developers

can determine and extract the features of the opponent behavior that generate excitement for the player, a mathematical formula can be designed in order to collectively represent them. Finally, the human players' experiment presented in Chapter 7 can be replicated to cross-validate and test variants of the proposed entertainment values.

9.3.2 Learning Methodology

The proposed on-line evolutionary learning method may also be successfully applied to any game during active real-time player-opponent interactions. Extracted features of this interaction may be used in order to estimate the fitness of the involved opponents according to their tasks. The replacement of the worst-fit opponent(s) method may be applied in frequent game periods to enhance the group's fitness. See also Stanley et al. (2005) for a successful application of this method in the NERO game. Moreover, on top of motion controllers additional game features could be also encoded in the evolutionary procedure and evolved on-line for optimizing entertainment. These features could include, in predator/prey games for instance, the speed of the opponents or the time in which the *Ghosts* are edible in a version of Pac-Man that includes power-pills.

Artificial evolution can explore complex search/state spaces efficiently and when combined with NNs it can demonstrate fast adaptability to dynamic and changing environments. Therefore neuro-evolution is recommended for learning in real-time. However, convergence time and unpredictability of the emergent behaviors constitute the disadvantages of the methodology which can be dealt with by careful design of the learning mechanism. Player modeling techniques are able to decrease convergence time down to realistic periods of time (i.e. tens of games) and furthermore proliferate the efficiency and justifiability of learning in real-time. Moreover, other efficient learning mechanisms such as temporal-difference learning (Tesauro, 2002) could also be applied to adjust neuro-controllers either for the motion or other levels of NPCs' control.

9.3.3 Controller

Artificial neural networks serve successfully the adaptability requirements for predator/prey reactive games in real-time. The ECWAS neuro-evolution approach is also able off-line to provide minimal ANN structures capable of achieving high performance behaviors. However, as the complexity of the opponents' tasks increases there

might be a need for more sophisticated structures of distributed representation. Memory of previous behaviors learned through the player-opponent interaction may very well be essential when a combination of various tasks is required. Recurrent NNs or augmented NN topologies with hidden states (Stanley and Miikkulainen, 2002) may be more appropriate when the opponents' tasks proliferate. Moreover, a hierarchy design of neuro-controllers that serve different opponent tasks could also provide the on-line learning mechanism with more flexibility and faster adaptability.

9.4 Summary

This dissertation has presented an efficient method for measuring player entertainment in predator/prey games. Moreover, it has provided a generic methodology for generating high entertainment values for such games. This methodology is grounded in adaptive evolutionary learning in real-time which is built on cooperative opponents of minimal controllers. Both the interest metric and the methodology proposed exhibit features of extensibility in other genres of games.

Appendix A

Bootstrapping

A.1 Interest Confidence Intervals

In order to minimize the non-deterministic effect of the player's strategy (whether that is hand-programmed or human) on the game's generated interest value as well as to draw a clear picture of this value's distribution, we apply the following bootstrapping procedure. Using a uniform random distribution we pick 10 different N-tuples out of $2N$ games. These 10 samples of data from N games are used to determine the game's interest value which is obtained from the average I value over the 10 samples. Moreover, the interest value's 95% confidence intervals $[I_l, I_u]$ are determined by the minimum and maximum of the 10 I values (Efron and Tibshirani, 1993).

A.2 Opponents' Performance

The bootstrapping procedure described in Section A.1 is also used to determine the 95% confidence intervals of the performance value P of a team of game opponents.

Appendix B

FlatLand: Tools and Experiments

B.1 Beta Distribution

When it is assumed that $\alpha+\beta$ independent experiments of an approach A experience α successes and β failures, the distribution of success rate p can be approximated with a Beta distribution. The Beta probability density function is given by:

$$f(\alpha,\beta,p) = \frac{1}{B(\alpha+1,\beta+1)} p^{\alpha}(1-p)^{\beta} \tag{A.1}$$

where $B(\alpha+1,\beta+1) = \int_0^1 p^{\alpha}(1-p)^{\beta}dp = \frac{\alpha!\beta!}{(\alpha+\beta+1)!}$. Assume that a random variable X with a beta-distribution has the upper bound χ_u and the lower bound χ_l for confidence limits such that $P(\chi_l < X < \chi_u) = 1-\varepsilon$ and $P(X \leq \chi_l) = \varepsilon/2$ where ε is the confidence coefficient (in all results presented here $\varepsilon = 0.05$). Then, we can assert that $[\chi_l,\chi_u]$ is a $(1-\varepsilon)\cdot 100$ percent confidence interval. If a probability p is beta distributed the confidence limits χ_l,χ_u can be obtained by solving the equations:

$$\frac{\varepsilon}{2} = \int_0^{\chi_l} \frac{p^{\alpha}(1-p)^{\beta}}{B(\alpha+1,\beta+1)} dp \tag{A.2}$$

$$\frac{\varepsilon}{2} = \int_{\chi_u}^1 \frac{p^{\alpha}(1-p)^{\beta}}{B(\alpha+1,\beta+1)} dp \tag{A.3}$$

From the lower and upper bound probability χ_l,χ_u, the 95% confidence effort cost can be estimated with $\left[\frac{1}{\chi_u}Q_A, \frac{1}{\chi_l}Q_A\right]$, where Q_A is the unit computing cost per run of the approach A (i.e. in our experiments Q_A equals to CPU time). Finally, the mean effort cost can be obtained from $\frac{\alpha+\beta+1}{\alpha}Q_A$.

B.2 Controller Size Experiment: BP

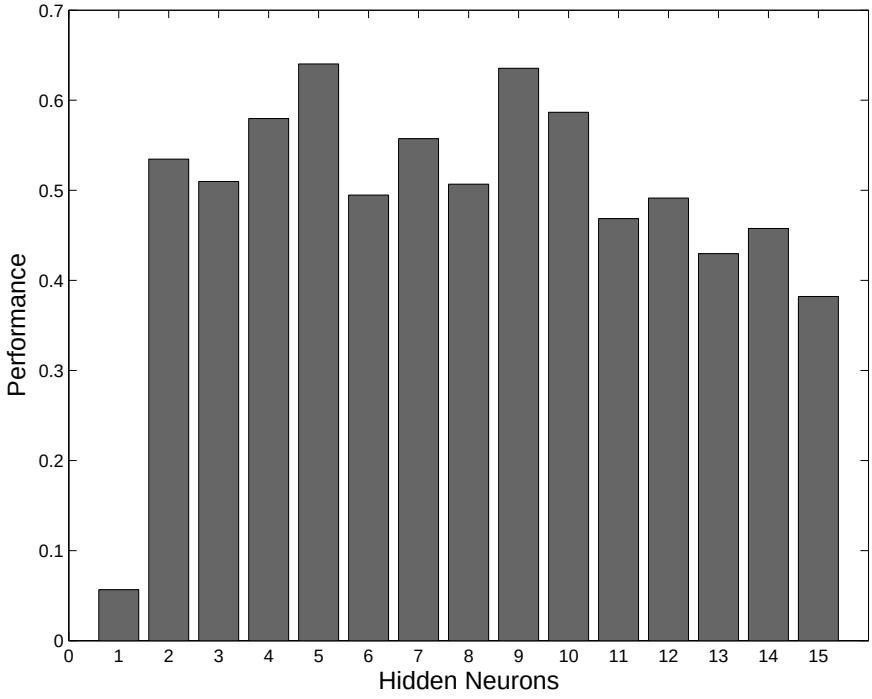
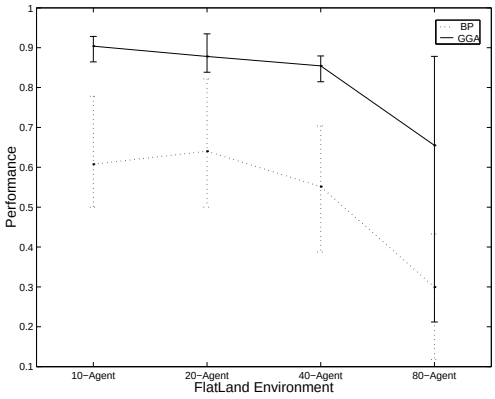


Figure B.1: Average performance (over 10 trials) of BP training in the 20-agent *FlatLand* environment.

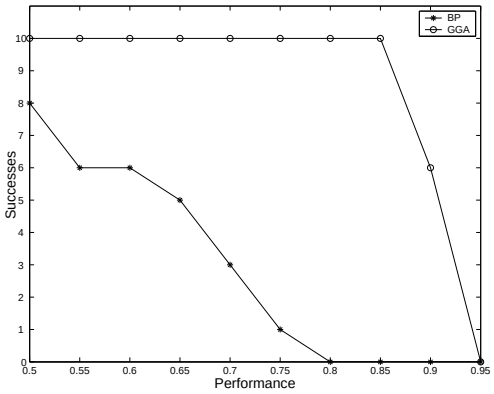
Hidden Neurons	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
σ^2 $(\cdot 10^{-2})$	0.1	0.3	2.0	12.3	1.6	20.3	9.3	10.1	4.2	8.5	2.3	6.1	9.3	15.9	13.2

Table B.1: Variance (over 10 trials) of BP training in the 20-agent *FlatLand* environment.

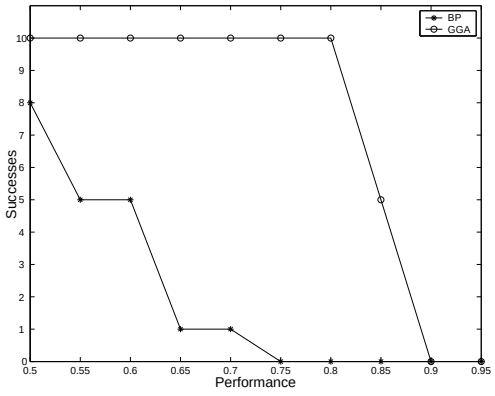
B.3 Growing FlatLand



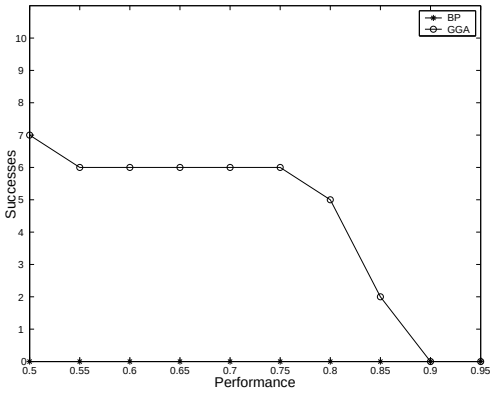
(a) Mean, best and worst performance of BP and GGA.



(b) 10-Agent *FlatLand* Environment.



(c) 40-Agent *FlatLand* Environment.



(d) 80-Agent *FlatLand* Environment.

Figure B.2: Increasing *FlatLand* complexity. (a): performance of BP and GGA over the *FlatLand* increasing population; (b), (c) and (d): number of successes out of 10 runs for specific performance values for both mechanisms.

[1] in seconds

Learning Mechanism	Parameters
BP	$S_t = 666$ (Animal path data)
GGA	$N_p = 20$, $N_s = 10\%$, $p_m = 0.01$ $e_p = 1000$ (N = 10), $e_p = 200$ (N = 40), $e_p = 150$ (N = 80), $\mathbf{T} = 2000$

Table B.2: Experiment parameters for the 10, 40 and 80-agent environments.

Environment	$P_{threshold}$	Approach	α	β	Confidence Interval	Effort Cost Interval[1]	Mean Effort Cost[1]
10-Agent	0.7	BP	3	7	[1.6401, 9.149]	[153.48, 856.17]	343.1267
		GGA	10	0	[1.0023, 1.3984]	[458.33, 639.46]	503.0080
	0.75	BP	1	9	[2.4225, 43.8596]	[226.70, 4104.39]	1029.3800
		GGA	10	0	[1.0023, 1.3984]	[458.33, 639.46]	503.0080
40-Agent	0.7	BP	1	9	[2.4225, 43.8596]	[226.70, 4104.39]	1029.3800
		GGA	10	0	[1.0023, 1.3984]	[765.59, 1068.14]	840.2130

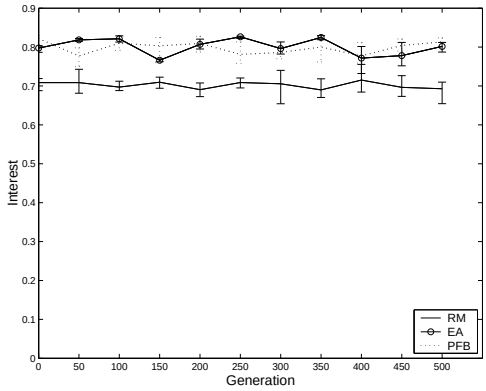
Table B.3: Effort cost comparison table ($\varepsilon = 0.05$) $Q_{\mathrm{BP}} = 93.58sec$, $Q_{\mathrm{GGA}} = 457.28sec$ for the 10-agent environment; $Q_{\mathrm{GGA}} = 763.83sec$ for the 40-agent environment.

Appendix C

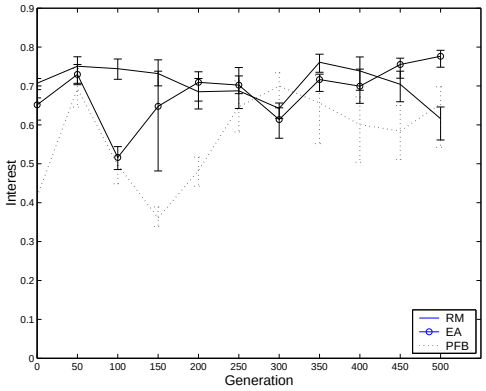
Dead End Experiments

Initial Behavior		Stage	Playing Against: RM I	RM $I_u - I$	RM $I - I_l$	EA I	EA $I_u - I$	EA $I - I_l$	PFB I	PFB $I_u - I$	PFB $I - I_l$
RM	A	5	0.692	0.027	0.022	0.830	0.008	0.015	0.846	0.014	0.025
		4	0.715	0.040	0.031	0.821	0.008	0.009	0.813	0.010	0.011
		3	0.749	0.012	0.016	0.771	0.009	0.015	0.831	0.005	0.007
	D	5	0.759	0.011	0.021	0.763	0.024	0.021	0.720	0.015	0.031
		4	0.761	0.021	0.026	0.776	0.015	0.028	0.700	0.035	0.053
		3	0.772	0.022	0.022	0.719	0.026	0.042	0.636	0.034	0.027
EA	D	5	0.707	0.014	0.031	0.701	0.010	0.016	0.617	0.042	0.062
		4	0.655	0.017	0.029	0.770	0.007	0.004	0.583	0.023	0.028
		3	0.733	0.029	0.030	0.750	0.016	0.010	0.795	0.032	0.045
PFB	A	5	0.691	0.014	0.011	0.751	0.012	0.018	0.722	0.069	0.118
		4	0.712	0.030	0.021	0.795	0.010	0.012	0.727	0.021	0.036
		3	0.784	0.014	0.072	0.766	0.008	0.017	0.696	0.037	0.042
	D	5	0.685	0.028	0.023	0.744	0.007	0.006	0.635	0.030	0.020
		4	0.723	0.008	0.012	0.766	0.011	0.017	0.696	0.008	0.008
		3	0.722	0.012	0.012	0.737	0.005	0.017	0.658	0.035	0.037
Average				0.0225			0.0140			0.0321	
Max				0.0720			0.0417			0.1183	
Min				0.0077			0.0038			0.0055	

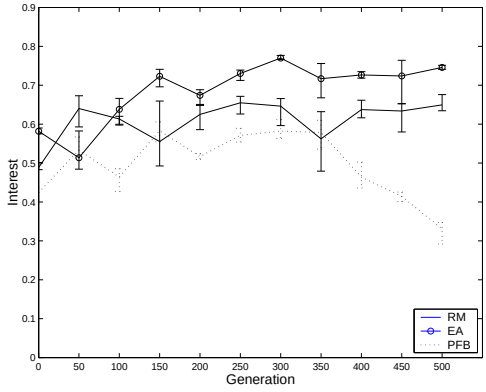
Table C.1: Confidence intervals of the interest values generated by on-line learning.



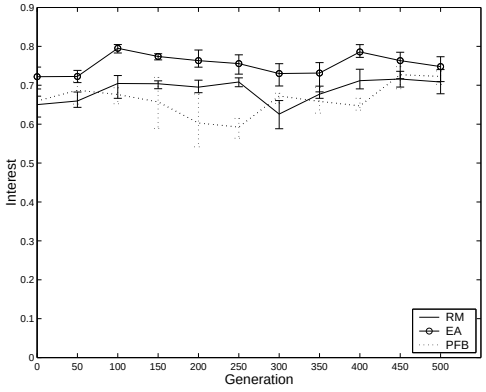
(a) RM Aggressive



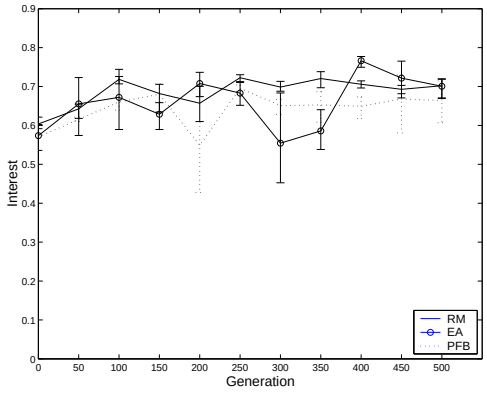
(b) RM Defensive



(c) EA Defensive

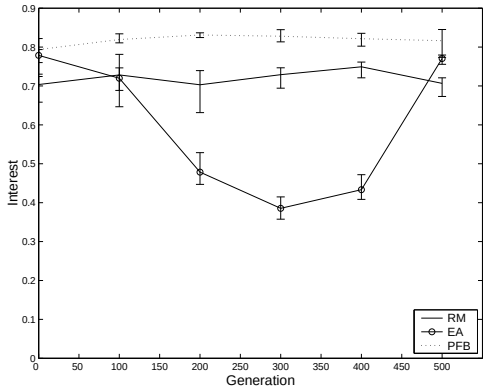


(d) PFB Aggressive

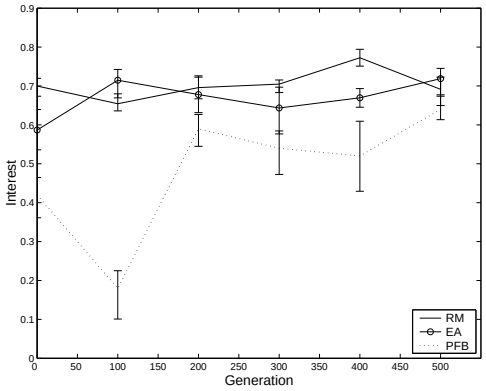


(e) PFB Defensive

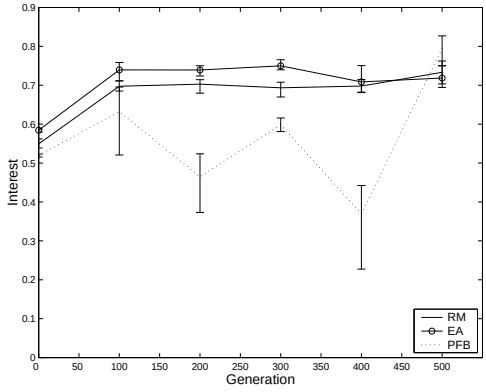
Figure C.1: Game interest over the number of OLL generations in the 4 *Dog* environment. Sub-figure captions denote the initial *Dogs*' behavior. Experiment Parameters: $e_p = 25$ simulation steps, $p_m = 0.02$, 5-hidden neurons controller.



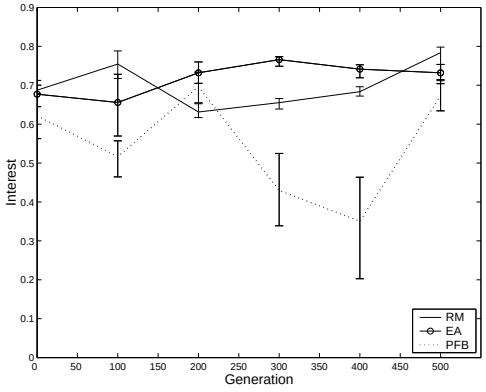
(a) RM Aggressive



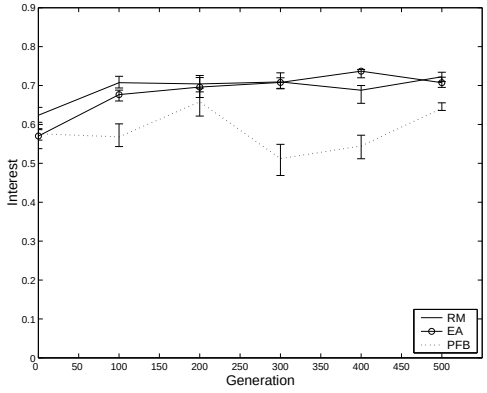
(b) RM Defensive



(c) EA Defensive



(d) PFB Aggressive



(e) PFB Defensive

Figure C.2: Game interest over the number of OLL generations in the 3 *Dog* environment. Sub-figure captions denote the initial *Dogs*' behavior. Experiment Parameters: $e_p = 25$ simulation steps, $p_m = 0.02$, 5-hidden neurons controller.

Appendix D

Pac-Man Experiments

		Playing Against								
		CB			RB			ADV		
Initial Behavior	Stage	I	$I_u - I$	$I - I_l$	I	$I_u - I$	$I - I_l$	I	$I_u - I$	$I - I_l$
B	Easy A	0.458	0.019	0.017	0.555	0.012	0.015	0.476	0.032	0.028
	Easy B	0.498	0.032	0.041	0.559	0.012	0.030	0.503	0.026	0.018
	Normal	0.576	0.033	0.031	0.412	0.035	0.030	0.442	0.027	0.022
	Hard	0.509	0.014	0.022	0.636	0.029	0.020	0.524	0.027	0.059
A	Easy A	0.646	0.053	0.035	0.614	0.022	0.045	0.513	0.027	0.022
	Easy B	0.624	0.065	0.051	0.562	0.019	0.033	0.552	0.036	0.050
	Normal	0.655	0.031	0.039	0.652	0.035	0.040	0.555	0.025	0.017
	Hard	0.613	0.042	0.037	0.688	0.022	0.028	0.650	0.030	0.045
H	Easy A	0.297	0.021	0.036	0.476	0.048	0.046	0.357	0.049	0.022
	Easy B	0.215	0.034	0.029	0.470	0.024	0.017	0.401	0.047	0.038
	Normal	0.190	0.021	0.019	0.250	0.026	0.048	0.423	0.042	0.055
	Hard	0.434	0.030	0.036	0.540	0.030	0.024	0.595	0.018	0.018
Average		0.0327			0.0287			0.0326		
Max		0.0647			0.0476			0.0586		
Min		0.0142			0.0119			0.0171		

Table D.1: Confidence intervals of the interest values generated by off-line training.

		Playing Against								
		CB			RB			ADV		
Initial Behavior	Stage	I	$I_u - I$	$I - I_l$	I	$I_u - I$	$I - I_l$	I	$I_u - I$	$I - I_l$
B	Easy A	0.819	0.021	0.020	0.771	0.028	0.032	0.566	0.017	0.049
	Easy B	0.763	0.026	0.016	0.747	0.023	0.023	0.661	0.010	0.012
	Normal	0.780	0.035	0.025	0.769	0.025	0.014	0.685	0.030	0.036
	Hard	0.739	0.011	0.010	0.756	0.014	0.017	0.746	0.041	0.052
A	Easy A	0.796	0.018	0.022	0.718	0.021	0.016	0.763	0.018	0.018
	Easy B	0.768	0.025	0.030	0.736	0.015	0.053	0.743	0.027	0.034
	Normal	0.753	0.014	0.011	0.741	0.014	0.019	0.802	0.015	0.016
	Hard	0.724	0.033	0.033	0.717	0.014	0.011	0.737	0.031	0.020
H	Easy A	0.762	0.022	0.031	0.822	0.011	0.019	0.737	0.046	0.053
	Easy B	0.772	0.009	0.016	0.771	0.023	0.027	0.709	0.021	0.021
	Normal	0.748	0.035	0.042	0.726	0.010	0.039	0.730	0.012	0.022
	Hard	0.742	0.019	0.016	0.762	0.018	0.018	0.788	0.011	0.023
Average			0.2290			0.0216			0.0270	
Max			0.0422			0.0537			0.0535	
Min			0.0097			0.0108			0.0103	

Table D.2: Confidence intervals of the best interest values generated by on-line learning.

			Playing In	
	PacMan Type	Behavior	Easy A	Easy B
OLT in Easy A	CB	B	0.9846	0.8854
		A	0.6999	0.6848
		H	0.5490	0.4940
	RB	B	0.7433	0.7382
		A	0.6623	0.7174
		H	0.5039	0.4916
	ADV	B	0.9040	0.8936
		A	0.7639	0.6187
		H	0.4638	0.5471
OLT in Easy B	CB	B	0.8512	0.8934
		A	0.7353	0.7873
		H	0.5012	0.5048
	RB	B	0.6947	0.7487
		A	0.6811	0.6945
		H	0.6152	0.5217
	ADV	B	0.8187	0.8764
		A	0.6389	0.6535
		H	0.2743	0.4179
Average			0.6714	0.6761
Variance			0.0295	0.0240

Table D.3: Easy A versus Easy B; performance comparison table.

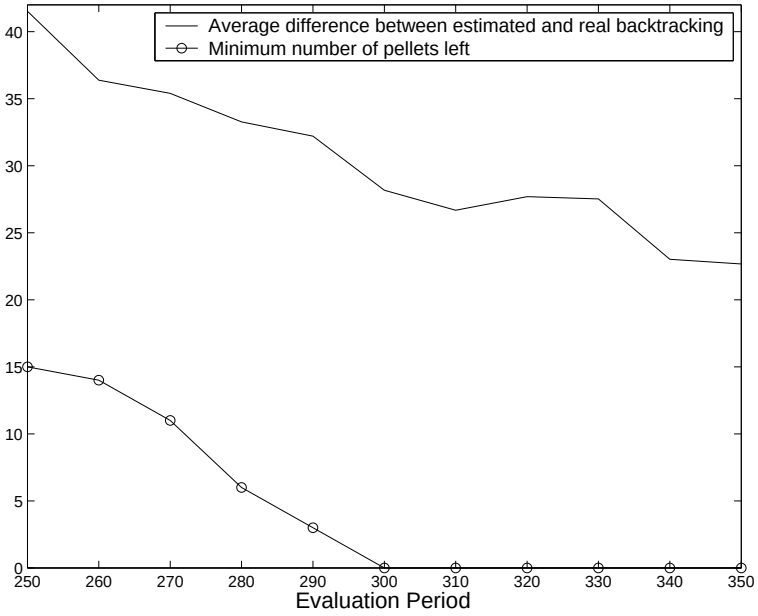


Figure D.1: Backtracking plot: the RB *PacMan* plays in the Easy stage without *Ghosts*. The average (over 100 games) difference between the estimated and real backtracking is obtained through the formula: Evaluation Period - (number of pellets eaten + backtrack movements).

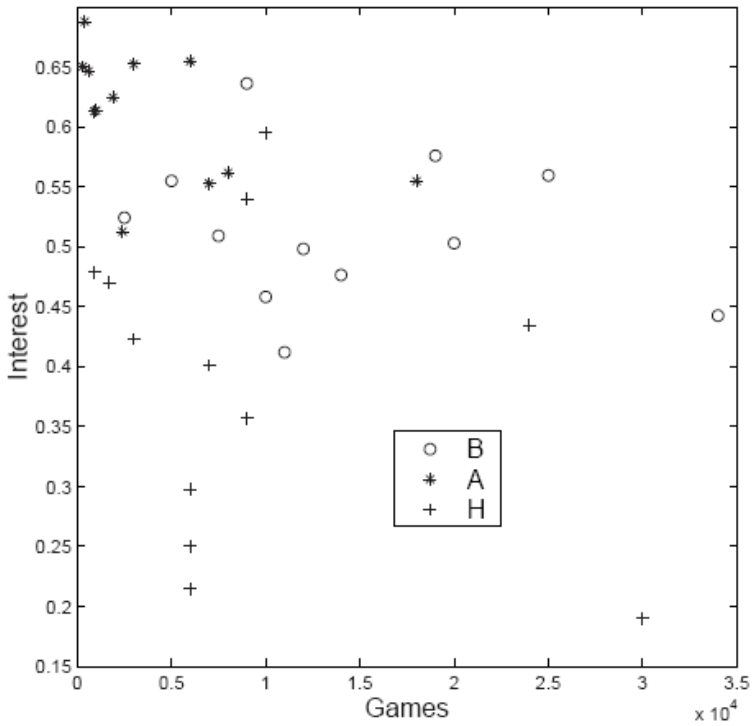
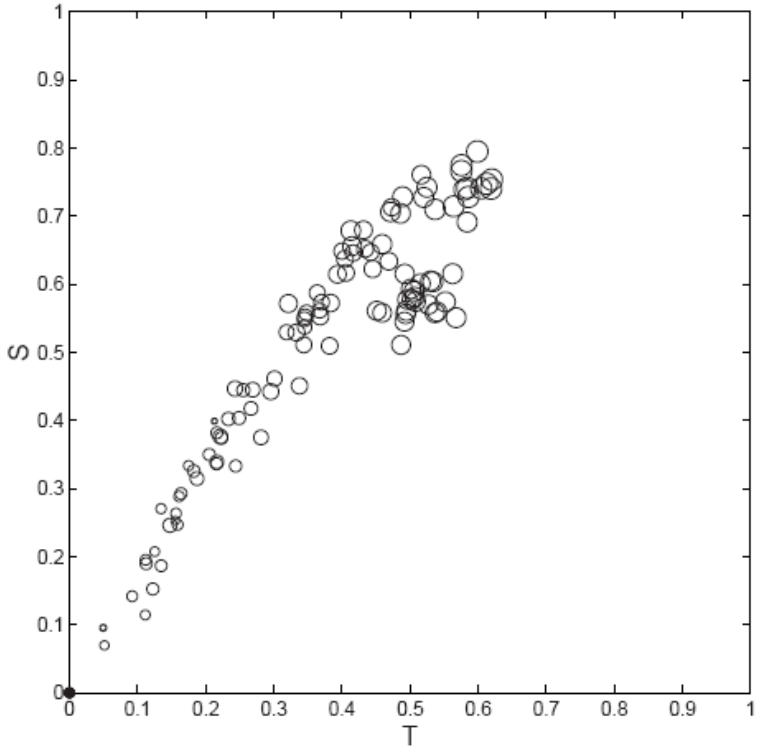
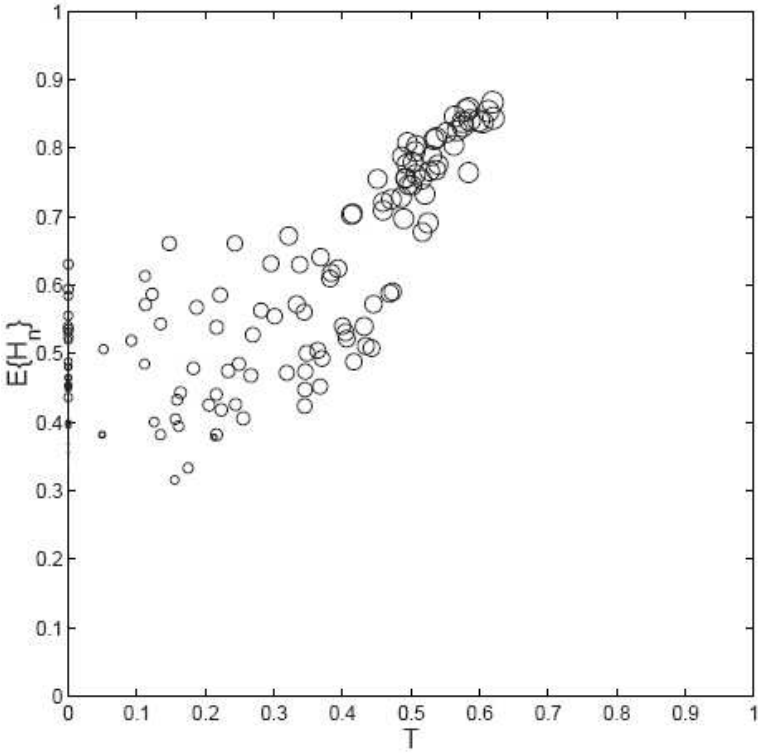


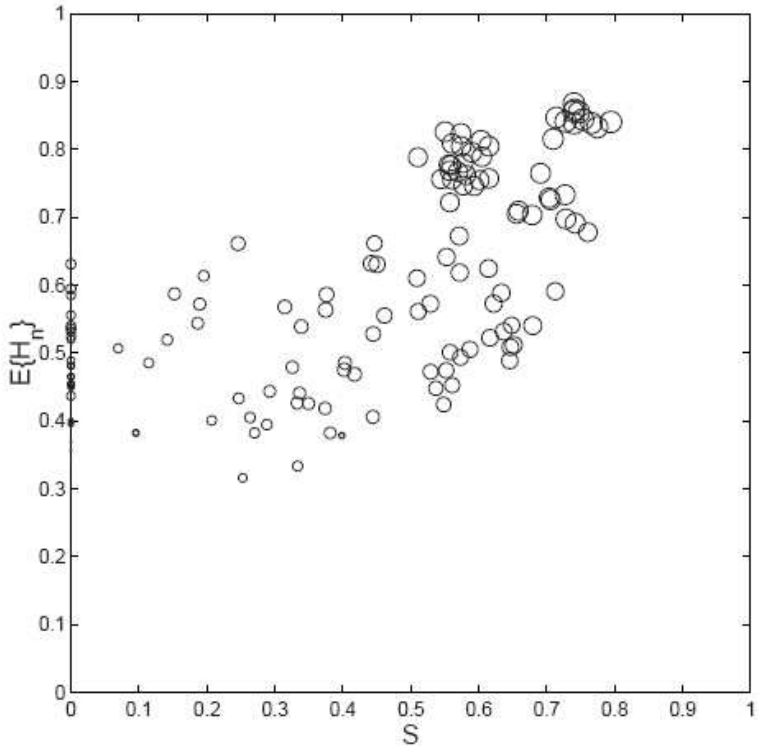
Figure D.2: Scatter plot of the initial I values over the games that OLL generated the highest interest value. Different shapes of data points indicate the initial OLT behaviors (B, A, H).



(a) S over T



(b) $E\{H_n\}$ over T



(c) $E\{H_n\}$ over S

Figure D.3: Scatter plots of the T, S and $E\{H_n\}$ values obtained during OLL in the Normal stage against the ADV *PacMan*. Three experiments are held with B, A and H initial *Ghost* behavior. The size of the data points is proportional to the generated I value.

Appendix E

Human Survey

GENERAL GUIDELINES (Please Read Carefully)

The computer game presented here is a modified version of the well-known Pac-Man arcade game released by Namco (Japan). Your objective as a player (appearing as yellow circle) is to get the highest score possible by eating pellets (small black squares) in a a maze-shaped stage while avoiding being killed by 4 opponent characters named 'Ghosts' (see Figure E.1 on next page). Ghosts attempt to kill you by touching you. The game is over when either all pellets in the stage are eaten by you (Pac-Man) or Ghosts manage to kill you or you are able to survive for a long period without eating all pellets from the stage. In that case, the game restarts from the same initial positions for you and the Ghosts. In this version of the game the stage's layout doesn't change even if the player manages to clear all the pellets from the stage.

The game is played by using the 4 arrow keys (up, down, left, right) to control the Pac-Man's motion on the stage. There are 2 functional buttons and a slider bar on the game:

Play!: Click this button every time you want to start playing a game. **CAUTION: VERY IMPORTANT!:** After clicking the **Play!** button you **must** double-click anywhere on the Pac-Man stage in order for you to start controlling Pac-Man (see Figure E.1).

Load Next Game: When a game is over the screen freezes. In order to continue playing, you have to click the **Load Next Game** button for the next game to be loaded. This button also pops-up useful instruction messages in between games (see Figure E.1).

Speed Bar: You can change the speed of the game by adjusting the vertical slider bar appearing at the left of the stage. In order to **increase** the speed of the game, drag the indicator **down**. If you want to **decrease** the speed of the game, drag the indicator **up** (see Figure E.1).

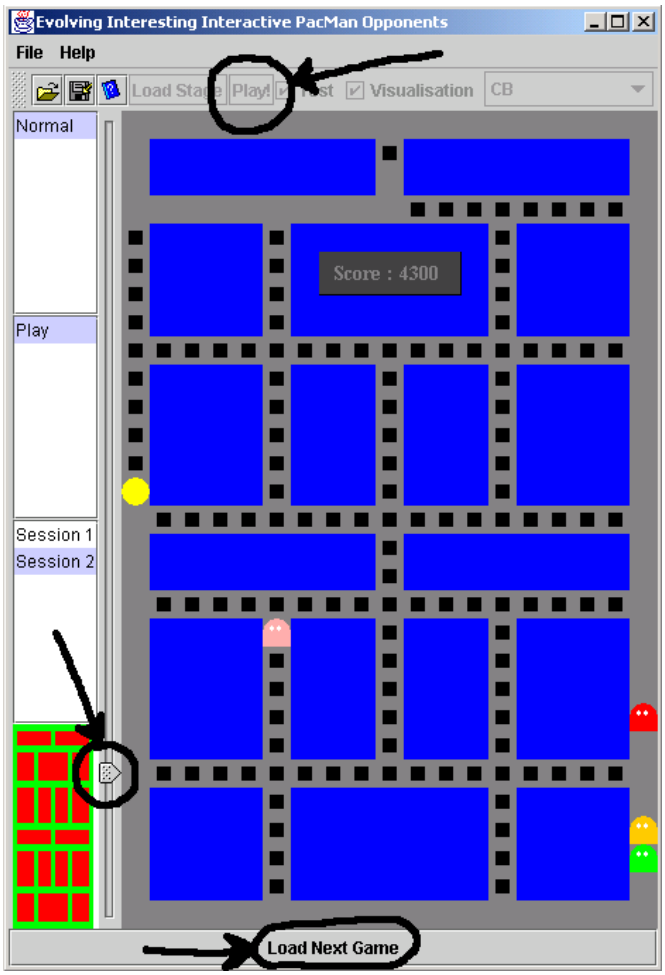


Figure E.1: Snapshot of the Pac-Man game. The 2 functional buttons and the slider are circled.

Subject Code:

FULL NAME:

e-mail:

Questionnaire

Please take the time to complete the following tasks, which will take you approximately 30 minutes to complete.

PART A

1. Do you like playing computer games?

Yes, I love them!	
Yes, but I'm not that enthusiastic about them.	
No, I don't.	

2. Do you like playing Pac-Man?

Yes, I am a fanatic player!	
Yes, I like it.	
Yes, but I'm not that enthusiastic about it.	
Yes, I used to like it, but not any more.	
No, I don't particularly like it.	

3. What factors do you consider make a good Pac-Man game? Please list them.

4. Now it's time for you to start playing Pac-Man! Take some time to familiarize yourself with it. In order to do that, click the **Play!** button and play for a testing

period (suggestion: this is a good chance for you to familiarize yourself with the speed of the game since it will be default for the games to follow in Part B once the testing period is over). The screen will freeze when the predefined testing period is over. When this occurs, proceed to the next page (Part B).

PART B

In this part you will play 12 short Pac-Man games in pairs (6 pairs). After completing each pair of games, a message on screen will ask you to complete a question (B.1-B.6) about this pair. Proceed to **Part B** by executing the following command:

Click the **Load Next Game** button and follow the instructions on screen. (Each time a game is over the screen will be freezed. In order to start a new game, you will have to click the **Load Next Game** button)

QUESTIONS (Please circle either YES or NO)

B.1 Was game no. 1 more interesting than game no. 2? YES NO

B.2 Was game no. 3 more interesting than game no. 4? YES NO

B.3 Was game no. 5 more interesting than game no. 6? YES NO

B.4 Was game no. 7 more interesting than game no. 8? YES NO

B.5 Was game no. 9 more interesting than game no. 10? YES NO

B.6 Was game no. 11 more interesting than game no. 12? YES NO

Please have a short **break** before going to the final part (**Part C**) of the questionnaire. Let Georgios know that you are having this break.

PART C

In this part you will play 2 pairs of Pac-Man games. These games will be a bit longer than the games played in **Part B**. After completing each pair of games, a message on screen will ask you to complete a question about this pair (please have a look at questions **C.1** and **C.2** before playing). Click **Play!** to start playing the first game and when the game is over (freezed screen) execute the following command:

Click the **Load Next Game** button and follow the instructions on screen. (Each time a game is over the screen will be freezed. In order to start a new game, you will have to click the **Load Next Game** button)

QUESTIONS (Please circle either YES or NO)

C.1 Was game no. 1 more interesting than game no. 2? YES NO

C.2 Was game no. 3 more interesting than game no. 4? YES NO

List the criteria you used for your assessment of which game is the more interesting:

Thanks for your cooperation! Enjoy your beer!

E.1 Experiment Tables

Subject No.	Factors of a good Pac-Man game[1]		
1	Intelligent Ghosts	Challenge	
2	Ghosts' Behavior		
3	Nice Graphics/Stages Topology/Sound		
4	Ghosts' Behavior		
5	Intelligent Ghosts	Nice Graphics/Stages Topology/Sound	Win 3/4 times
6	Intelligent Ghosts	Speed	
7	Nice Graphics/Stages Topology/Sound	Diverse Ghosts' behavior	
8	Challenge		
9	Nice Graphics/Stages Topology/Sound	Challenge	
10	Nice Graphics/Stages Topology/Sound	Challenge	Intelligent Ghosts
11	Nice Graphics/Stages Topology/Sound	Speed	
12	Unpredictable Ghosts	Challenge	Adaptability
13	Challenge		
14	Nice Graphics/Stages Topology/Sound		
15	Nice Graphics/Stages Topology/Sound	Challenge	Game Controls
16	Nice Graphics/Stages Topology/Sound		
17	Intelligent Ghosts	Challenge	
18	I don't know		
19	My Moods		
20	Nice Graphics/Stages Topology/Sound	Game Controls	
21	Be different from the Original Version		
22	Nice Graphics/Stages Topology/Sound		
23	Nice Graphics/Stages Topology/Sound		
24	Nice Graphics/Stages Topology/Sound	Challenge	Speed
25	Nice Graphics/Stages Topology/Sound	Game Controls	
26	Nice Graphics/Stages Topology/Sound	Power Pills	
27	Intelligent Ghosts	Speed	Game Controls
28	Nice Graphics/Stages Topology/Sound	Cooperative Ghosts	Diverse Ghosts' behavior
29	Nice Graphics/Stages Topology/Sound	Adaptability	Interface
30	I don't know		

Table E.1: Answers on A.3.

[1] Answers in this table are either presented as written by the subjects or alternatively they are coded to match specific categories of factors. The categories are: Intelligent Ghosts, Unpredictable Ghosts, Ghosts' behavior, Adaptability, Challenge, Diverse Ghosts' behavior, Nice Graphics/Stages Topology/Sound, Game Controls and Speed.

[2] Answers in this table are either presented as written by the subjects or alternatively they are coded to match specific categories of criteria. The categories are: 1) Opponent Oriented: Intelligent Ghosts, Unpredictable Ghosts, Ghosts' behavior, Challenge, Diverse Ghosts' behavior, Spatial Diversity and 2) Non-Opponent Oriented: Performance, I don't know/Randomness, Nice Graphics/Stages Topology/Sound, Game Controls, Speed.

Subject No.	Interest Criteria[2]		
1	Intelligent Ghosts		
2	Ghosts' Behavior		
3	Ghosts' Behavior	Spatial Diversity	Challenge
4	Ghosts' Behavior		
5	Performance	Challenge	
6	Game Controls	Challenge	
7	Ghosts' Behavior	Spatial Diversity	Challenge
8	Ghosts' Behavior	Spatial Diversity	Challenge
9	Performance	Challenge	
10	Performance		
11	Unpredictable Ghosts		
12	Performance	Challenge	
13	I don't know/Randomness		
14	Game Controls		
15	Challenge		
16	Speed		
17	Challenge		
18	Intelligent Ghosts		
19	Game Controls	Performance	
20	Ghosts' Behavior	Spatial Diversity	Challenge
21	Challenge		
22	I don't know/Randomness		
23	Nice Graphics/Stages Topology/Sound		
24	Challenge		
25	Game Controls		
26	Challenge		
27	I don't know/Randomness	Game Controls	
28	Ghosts' Behavior		
29	Challenge	Diverse Ghosts' behavior	I don't know/Randomness
30	I don't know/Randomness		

Table E.2: Answers on C.2.

	Opponent															
Subject No.	Part B												Part C			
1	1	2	3	1	3	2	2	1	2	3	1	3	OLL_1	4	4	OLL_2
2	4	2	1	4	1	2	2	4	2	1	4	1	OLL_1	4	4	OLL_2
3	2	1	5	1	5	2	1	5	2	5	1	2	OLL_1	4	4	OLL_2
4	1	3	4	3	3	1	4	1	3	4	1	4	OLL_1	4	4	OLL_2
5	5	3	1	5	1	3	3	5	3	1	5	1	OLL_1	4	4	OLL_2
6	4	1	4	5	1	4	5	1	5	4	1	5	4	OLL_1	OLL_2	4
7	3	2	2	4	2	3	3	4	4	2	4	3	4	OLL_1	OLL_2	4
8	2	3	3	5	2	5	5	3	3	2	5	2	4	OLL_1	OLL_2	4
9	5	2	2	5	4	2	4	5	2	4	5	4	4	OLL_1	OLL_2	4
10	3	4	5	4	5	3	4	3	3	5	4	5	4	OLL_1	OLL_2	4

Table E.3: Set of games' sequence assigned for subjects of each type.

	Opponent																
Subject No.	Part B												Part C				Mean
1	3870	4380	4220	4310	3950	5020	4290	3200	4000	3680	3310	3970	29570	31730	34170	31930	192145.8
2	3490	4380	4190	2710	3990	4340	3950	3430	3960	3600	2010	1350	27600	18880	22770	23600	
3	5490	5430	3450	3800	4990	5270	5520	4780	5060	4150	5690	5650	28840	30970	32010	34480	
4	4746	3348	3654	1368	3156	4764	4086	4818	3408	2952	4794	4674	22420	29990	26220	27980	
5	4850	4630	4270	3890	3990	4120	2950	3810	3440	4060	4970	4090	27560	27220	29500	25910	
6	1510	2920	2420	2810	2240	2310	2610	2600	2490	2530	1780	1790	25130	23160	23130	24340	
7	3520	4210	4260	3810	4840	3070	2790	2830	2950	4330	4140	3420	29630	28780	24370	27690	
8	4420	3480	1270	2780	3850	3600	4440	4350	3600	4540	3480	4700	20450	20790	22170	25220	
9	4330	4180	4440	3130	3500	3240	2770	2400	4540	2050	3280	3110	26630	29660	33340	32290	
10	2420	30	2520	1580	3200	2040	590	3070	2850	2380	30	1470	21560	18060	17210	15720	
11	3490	2540	2290	3440	2910	3530	2700	4040	4210	2070	4040	2010	31610	20730	25190	31610	170088
12	390	3740	3960	2360	3260	4140	2210	2010	4280	4250	2720	2460	19700	23810	26150	19760	
13	2940	3530	3450	2970	4050	2770	3560	2450	4100	2540	4550	3530	32340	37240	31430	33040	
14	3640	4230	2390	4800	1050	4080	4030	4400	2550	2580	2730	3480	21810	25220	23280	25930	
15	4460	3090	3300	2430	4200	4160	3700	4510	2600	3420	3300	4080	24290	25920	25870	22810	
16	1560	2970	2970	2670	3480	2180	1850	3720	3630	2180	3550	2880	18110	19320	18780	20180	
17	1730	3790	4280	2250	3930	2980	2230	3370	2640	3140	3150	3830	29530	28520	24870	26680	
18	2620	2370	1670	1530	2850	3140	2650	1220	3030	2550	3180	1760	20390	18480	19540	23390	
19	2020	3440	2380	2400	3510	2140	2560	1970	3390	2700	2650	2040	23810	20430	21330	26390	
20	1840	3150	870	970	2310	2480	2370	3280	3590	2420	3570	3180	25540	25180	28280	31540	157551
21	3530	3390	1980	2780	4100	1570	2730	2440	4250	3070	2710	3150	24020	28030	24730	26850	
22	1800	3120	3090	3260	1700	3090	2400	650	3650	3180	2130	2420	19970	18300	22530	24150	
23	3740	4680	3480	5560	2570	3580	3550	3300	4460	3570	4540	3870	22970	27730	26640	25760	
24	3370	2580	2410	2250	3480	3610	1940	2860	3580	1910	3650	3110	24580	26100	23260	27010	
25	1520	1850	1230	820	2420	1670	2880	2240	3230	1910	1610	2110	20690	21190	21480	21970	
26	960	1770	3080	1590	680	2310	1720	2510	2830	1270	2390	2370	21930	20910	20260	19730	
27	1770	2380	1210	820	2410	1600	2000	1930	1560	1670	3090	1240	21440	17620	17990	12800	
28	2170	2480	1360	1870	4320	1160	4170	2010	3700	3720	2540	3590	19850	20260	20820	25770	
29	4100	4680	4880	2880	3880	5360	3220	4360	5200	2550	4900	3720	29160	31180	27030	27950	
30	1730	1550	1930	1980	970	2290	1410	1500	2900	1850	1700	890	15830	14420	17920	18860	

Table E.4: Subjects' scores ranked by subject type; i.e. 1–10: Like, 11–20: Neutral, 21–30: Don't Like.

Bibliography

Edwin A. Abbott. *Flatland*. Signet Classic, June 1984.

D. H. Ackley and M. L. Littman. Interactions between learning and evolution. In C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, editors, *Artificial Life II*, pages 478–507, Reading, MA, 1992. Sante Fe Institute Studies in the Sciences and Complexity, Addison-Wesley.

P. J. Angeline, G. M. Sauders, and J. B. Pollack. An evolutionary algorithm that constructs recurrent neural networks. *IEEE Trans. Neural Networks*, 5:54–65, 1994.

T. Bäck and H. P. Schwefel. An overview of evolutionary algorithms for paramater optimization. *Evolutionary Computation*, 1(1):1–23, 1993.

T. Balch and R. C. Arkin. Behavior-based formation control for multi-robot teams. *IEEE Transactions on Robotics and Automation*, 14(6):926–939, December 1998.

C. Bauckhage, C. Thurau, and G. Sagerer. Learning human-like opponent behavior for interactive computer games. *Pattern Recognition, Lecture Notes in Computer Science 2781*, pages 148–155, 2003.

BBC. Computer games are good for you! BBC News web-site, 18 March 2002. http://news.bbc.co.uk/cbbcnews/hi/uk/newsid_1879000/1879361.stm.

BBC. Video games 'stimulate learning'. BBC News web-site, 18 March 2002. http://news.bbc.co.uk/1/hi/education/1879019.stm.

C. Beal, J. Beck, D. Westbrook, M. Atkin, and P. Cohen. Intelligent modelling of the user in interactive entertainment. In *Proceedings of the AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment*, pages 8–12, Stanford, 2002.

Richard K. Belew, John McInerney, and Nicol N. Schraudolph. Evolving networks:

Using the genetic algorithm with connectionist learning. In Christopher G. Langton, Charles Taylor, J. Doyne Farmer, and Steen Rasmussen, editors, *Artificial Life II*, pages 511–547. Addison-Wesley, Redwood City, CA, 1992.

Endika Bengoetxea, Pedro Larranaga, Isabelle Bloch, and Aymeric Perchant. Image recognition with graph matching using estimation of distribution algorithms. In *Proceedings of Medical Image Understanding and Analysis*, pages 89–92, Birmingham, UK, 2001.

Y. Bjorsson, V. Hafsteinsson, A. Johansson, and E. Jonsson. Efficient use of reinforcement learning in a computer game. In *Proceedings of the International Conference on Computer Games: Artificial Intelligence, Design and Education*, pages 379–383, Microsoft Campus, Reading, UK, November 2004.

A.D. Blair and E. Sklar. Exploring evolutionary learning in a simulated hockey environment. In *Congress on Evolutionary Computation*, pages 197–203. IEEE Service Center, 1999.

Bruce Blumberg, Marc Downie, Yuri Ivanov, Matt Berlin, Michael Patrick Johnson, and Bill Tomlinson. Integrated learning for interactive synthetic characters. In *Proceedings of ACM SIGGRAPH 2002*, 2002.

Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, New York, 1999.

R. Boon. The 40 hour millstone. *Computer Trade Weekly*, (877), February 2002.

David M. Bourg. *Physics for Game Developers*. O' Reily and Associates, Sebastopol, CA, 2001.

Rodney A. Brooks. Elephants don't play chess. *Robotics and Autonomous Systems*, 6, 1990.

Jennifer Burg and Beth Cleland. Computer-enhanced or computer-enchanted: The magic and mischief of learning with computers. In *Proceedings of ED-MEDIA 2001, AACE*, June 2001.

S. Cass. Mind games. *IEEE Spectrum*, pages 40–44, 2002.

Alex J. Champandard. *AI Game Development*. New Riders Publishing, 2004.

D. Charles and M. Black. Dynamic player modelling: A framework for player-centric

digital games. In *Proceedings of the International Conference on Computer Games: Artificial Intelligence, Design and Education*, pages 29–35, 2004.

J. Cheng and R. Greiner. Learning bayesian belief network classifiers: Algorithms and system. In *Proceedings of the Canadian Conference on Artificial Intelligence*, 2001.

D. Cliff and S. Grand. The creatures global digital ecosystem. *Artificial Life*, 5:77–94, 1999.

Nicholas Cole, Sushil J. Louis, and Chris Miles. Using a genetic algorithm to tune first-person shooter bots. In *Proceedings of the 2004 Congress on Evolutionary Computation*, pages 139–145, 2004.

Robert J. Collins and David R. Jefferson. Antfarm: Towards simulated evolution. In Christopher G. Langton, Charles Taylor, J. Doyne Farmer, and Steen Rasmussen, editors, *Artificial Life II*, pages 579–601. Addison-Wesley, Redwood City, CA, 1992.

C. Crawford. Revolution and apocalypse. *Interactive Entertainment Design*, 7, 1994.

C. Crawford. Computer games are dead. *Interactive Entertainment Design*, 9, 1996.

Nick Crispini. Considering the growing popularity of online games: What contibutes to making an online game attractive, addictive and compeling. Dissertation, SAE Institute, London, October 2003.

Pedro Demasi and Adriano J. de O. Cruz. On-line coevolution for action games. In *Proceedings of the 3rd Interational Conference on Intelligent Games and Simulation (GAME-ON)*, pages 113–120, 2002.

Pedro Demasi and Adriano J. de O. Cruz. Online coevolution for action games. *International Journal of Intelligent Games & Simulation*, 2(2):80–88, 2003.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statist. Soc.*, (39(B)):1–38, 1977.

Heather Desurvire, Martin Caplan, and Jozsef A. Toth. Using heuristics to evaluate playability of games. In *CHI conference*, Vienna, Austria, 2004.

Jonathan Dinerstein, Parris K. Egbert, Hugo de Garis, and Nelson Dinerstein. Fast

and learnable behavioral and cognitive modeling for virtual character animation. *Computer Animation and Virtual Worlds*, 15(2), 2004.

M. Dorigo, G. Di Caro, and L. Gambardella. Ant colony optimization: A new meta-heuristic. In Peter J. Angeline, Zbyszek Michalewicz, Marc Schoenauer, Xin Yao, and Ali Zalzala, editors, *Proceedings of the Congress on Evolutionary Computation*, volume 2, pages 1470–1477. IEEE Press, 1999.

R. O. Duda and P. E. Hart. *Bayes Decision Theory*, chapter 2, pages 10–43. 1973.

P. Eckman. Facial expressions of emotions. *Annual Review of Psychology*, 20:527–554, 1979.

B. Efron and R. J. Tibshirani. *An introduction to the bootstrap*. New York: Chapman & Hall, 1993.

Electronic-Arts. Black and white, 2003. http://www.bwgame.com/.

J. L. Elman. Finding structure in time. *Cognitive Science*, 14:179–211, 1990.

Entertainment Software Association (ESA). ESA'S 2004 essential facts about the computer and video game industry. 2004. http://www.theesa.com/.

Oren Etzioni. Intelligence Without Robots (a reply to Brooks). *AI Magazine*, 14(4), 1993.

F. Flacher and O. Sigaud. Spatial coordination through social potential fields and genetic algorithms. In B. Hallam, D. Floreano, J. Hallam, G. Hayes, and J.-A. Meyer, editors, *From Animals to Animats 7. Proceedings of the Seventh International Conference on Simulation of Adaptive Behavior*, pages 389–390. The MIT Press, 2002.

F. Flacher and O. Sigaud. Basc, a bottom-up approach to automated design of spatial coordination. In S. Schaal, A. Ijspeert, A. Billard, Sethu Vijayakumar, J. Hallam, and J.-A. Meyer, editors, *From Animals to Animats 8. Proceedings of the Seventh International Conference on Simulation of Adaptive Behavior*, pages 435–444. The MIT Press, 2004.

L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial intelligence through simulated evolution*. Wiley, New York, 1966.

David B. Fogel, Timothy J. Hays, and Douglas R Johnson. A platform for evolving

characters in competitive games. In *Proceedings of the Congress on Evolutionary Computation (CEC-04)*, pages 1420–1426, June 2004.

David B. Fogel. Using evolutionary programming to construct neural networks that are capable of playing tic-tac-toe. In *Proceedings of the IEEE International Conference on Neural Networks*, pages 875–880, San Francisco, CA, 1993. IEEE Press.

David B. Fogel. *Blondie24: Playing at the edge of AI*. Morgan Kaufmann, 2002.

Colin M. Frayn. An evolutionary approach to strategies for the game of monopoly. In Graham Kendall and Simon Lucas, editors, *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, pages 66–72, Essex University, Colchester, UK, 4–6 April 2005.

M. Frean. The upstart algorithm: A method for concructing and training feedforward neural networks. *Neural Computation*, 2:198–209, 1990.

M. Freed, T. Bear, H. Goldman, G. Hyatt, P. Reber, A. Sylvan, and J. Tauber. Towards more human-like computer opponents. In *Working Notes of the AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment*, pages 22–26, 2000.

Pablo Funes and Jordan Pollack. Measuring progress in coevolutionary competition. In *From Animals to Animats 6: Proceedings of the* 6^{th} *International Conference on Simulation of Adaptive Behavior (SAB-00)*, pages 450–459. The MIT Press, 2000.

Pablo Funes, Elisabeth Sklar, Hugues Jullié, and Jordan Pollack. Animal-animat coevolution: Using the animal population as fitness function. In *From Animals to Animats 5: Proceedings of the* 5^{th} *International Conference on Simulation of Adaptive Behavior (SAB-98)*, pages 525–533. The MIT Press, 1998.

John D. Funge. *Artificial Intelligence for Computer Games*. A. K. Peters Ltd, 2004.

Marcus Gallagher and Amanda Ryan. Learning to play pac-man: An evolutionary, rule-based approach. In *Proceedings of the Congress on Evolutionary Computation (CEC)*, pages 2462–2469, 2003.

Z. Ganon, A. Keinan, and E. Ruppin. Evolutionary network minimization: Adaptive implicit pruning of successful agents. In W. Banzhaf, T. Christaller, P. Dittrich,

J. T. Kim, and J. Ziegler, editors, *Advances in Artificial Life - Proceedings of the 7th European Conference on Artificial Life (ECAL)*, pages 319–327, 2003.

D. E. Goldberg. *Computer-Aided Gas Pipeline Operation using Genetic Algorithms and Rule Learning*. Ph.D. dissertation, University of Michigan, 1983.

D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA, 1989.

C. González, J. A. Lozano, and P. Larrañaga. Mathematical modelling of UMDA$_c$ algorithm with tournament selection. Behaviour on linear and quadratic functions. *International Journal of Approximate Reasoning*, 31(3):313–340, 2002.

Thore Graepel, Ralf Herbrich, and Julian Gold. Learning to fight. In *Proceedings of the International Conference on Computer Games: Artificial Intelligence, Design and Education*, pages 193–200, Microsoft Campus, Reading, UK, November 2004.

R. Grzeszczuk, D. Terzopoulos, and G. Hinton. Neuroanimator: fast neural network emulation and control of physics-based models. In *Proceedings of the 25th annual Conference on Computer Graphics and Interactive Techniques*, pages 9–20. ACM Press, 1998.

M. T. Hagan and M. Menhaj. Training feedforward networks with the marquardt algorithm. *IEEE Transactions on Neural Networks*, 5(6):989–993, 1994.

W. Hartmann and B. Rollett. Positive interaction in the elementary school. pages 195–202, 1994.

Simon Haykin. *Neural Networks: A Comprehensive Foundation*. Macmillian College Publishing Company Inc., Upper Saddle River, NJ, USA, 2^{nd} edition, 1998.

Thomas Haynes and Sandip Sen. Evolving behavioral strategies in predators and prey. In Sandip Sen, editor, *IJCAI-95 Workshop on Adaptation and Learning in Multiagent Systems*, pages 32–37, Montreal, Quebec, Canada, 1995. Morgan Kaufmann.

D. O. Hebb. *The Organization of Behavior*. Wiley, New York, 1949.

D. Heckerman, D. Geiger, and D. M. Chickering. Learning bayesian networks: the combination of knowledge and statistical data. *Machine Learning*, 20(197–243), 1995.

J. Hertz, A. Krogh, and R. Palmer. *Introduction to the Theory of Neural Computation*. Addison-Wesley, Reading, MA, 1991.

G. E. Hinton and S. J. Nowlan. How learning can guide evolution. *Complex Systems*, 1:495–502, 1987.

John H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975.

E. Horvitz and M. Barry. Display of information for time-critical decision making. In *Proceedings of the 11th Conference on Uncertainty in AI*, pages 296–305, 1995.

R. Houlette. *Player Modeling for Adaptive Games. AI Game Programming Wisdom II*, pages 557–566. Charles River Media, Inc, 2004.

R. Le Hy, A. Arrigoni, P. Bessière, and O. Lebeltel. Teaching bayesian behaviors to video game characters. *Robotics and Autonomous Systems*, 47:177–185, 2004.

Icosystem. The Game, 2002. http://www.icosystem.com/game.htm.

Hiroyuki Iida, N. Takeshita, and J. Yoshimura. A metric for entertainment of boardgames: its implication for evolution of chess variants. In R. Nakatsu and J. Hoshino, editors, *IWEC2002 Proceedings*, pages 65–72. Kluwer, 2003.

D. Isla and B. Blumberg. New challenges for character-based AI for games. In *Proccedings of the AAAI Spring Symposium on AI and Interactive Entertainment*, pages 41–45. AAAI Press, 2002.

S. Johnson. *Adaptive AI*, pages 639–647. Charles River Media, Hingham, MA, 2004.

S. Kaiser and T. Wehrle. Situated emotional problem solving in interactive computer games. In N. H. Frijda, editor, *Proceedings of the VIXth Conference of the International Society for Research on Emotions*, pages 276–280. ISRE Publications, 1996.

S. Kaiser, T. Wehrle, and S. Schmidt. Emotional episodes, facial expressions, and reported feelings in human computer interactions. In A. H. Fisher, editor, *Proceedings of the Xth Conference of the International Society for Research on Emotions*, pages 82–86. ISRE Publications, 1998.

O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *Int. J. for Robotics Research*, 5(1):90–99, 1986.

A. Khoo and R. Zubeck. Applying inexpensive techniques to computer games. *IEEE Intelligent Systems*, pages 2–7, 2002.

DaeEun Kim. *A Quantitative Approach to the Analysis of Memory Requirements for Autonomous Agent Behaviours using Evolutionary Computation*. PhD thesis, University of Edinburgh, 2002.

T. Kohonen. *Self-Organizing Maps*, volume 1. Springer Verlag, 1997.

A. N. Kolmogorov. On the representation of continuous functions of several variables by superposition of continuous functions of one variable and addition. *Doklady Akademii. Nauk USSR*, 114:679–681, 1957.

K. B. Korb, A. E. Nicholson, and N. Jitnah. Bayesian poker. uncertainty in artificial intelligence, 1999. Stockholm, Sweden.

J. Koza. *Genetic Programming: On the programming of computers by means of natural selection*. MIT Press, 1992.

V. Kurkova. Kolmogorov's theorem is relevant. *Neural Computation*, 3:617–622, 1991.

John E. Laird and Michael van Lent. Human-level AI's killer application: Interactive computer games. In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI)*, pages 1171–1178, 2000.

John E. Laird. Research in human-level AI using computer games. *Communications of the ACM*, 3(8):32–35, 2002.

P. Larrañaga and J. A. Lozano. *Estimation of Distribution Algorithms. A new tool for Evolutionary Computation*. Kluwer Academic Publishers, 2001.

Daniel Livingstone and Stephen J. McGlinchey. What believability testing can tell us. In *Proceedings of the International Conference on Computer Games: Artificial Intelligence, Design and Education*, pages 273–277, Microsoft Campus, Reading, UK, November 2004.

Simon Lucas. Evolving a neural network location evaluator to play Ms. Pac-Man. In Graham Kendall and Simon Lucas, editors, *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, pages 203–210, Essex University, Colchester, UK, 4–6 April 2005.

Sean Luke and Lee Spector. Evolving teamwork and coordination with genetic programming. In John R. Koza, David E. Goldberg, David B. Fogel, and Rick L. Riolo, editors, *Genetic Programming 1996: Proceedings of the First Annual Conference*, pages 150–156, Stanford University, CA, USA, 1996. MIT Press.

John Manslow. *Learning and Adaptation*, pages 557–566. Charles River Media, Hingham, MA, 2002.

John Manslow. *Using Reinforcement Learning to Solve AI Control Problems*, pages 557–566. Charles River Media, Hingham, MA, 2002.

M. Maragoudakis, A. Thanopoulos, K. Sgarbas, and N. Fakotakis. Domain knowledge acquisition and plan recognition by probabilistic reasoning. *Int. Journal of Artificial Intelligence Tools, Special Issue in AI Techniques in Web-Based Educational Systems*, 13(2):333–365, 2004.

J. A. Meyer and A. Guillot. From SAB90 to SAB94: Four years of animat research. In *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior (SAB-92)*, 1994.

Chris Miles and Sushil Louis. Case-injection improves response time for a real-time strategy game. In Graham Kendall and Simon Lucas, editors, *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, pages 149–156, Essex University, Colchester, UK, 4–6 April 2005.

G. Miller and D. Cliff. Protean behavior in dynamic games: Arguments for the coevolution of pursuit-evasion tactics. In D. Cliff, P. Husbands, J.-A. Meyer, and S. W. Wilson, editors, *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior (SAB-94)*, pages 411–420, Cambridge, MA, 1994. MIT Press.

M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge MA, 1996.

T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

Peter Molynoeux. Postmortem: Lionhead studios' black and white. *Game Developer*, June 2001.

D. J. Montana and L. D. Davis. Training feedforward neural networks using genetic algorithms. In *Proceedings of the Eleventh International Joint Conference on*

Artificial Intelligence (IJCAI-89), pages 762–767, San Mateo, CA, 1989. Morgan Kauffman.

Heinz Muehlenbein and Gerhard Paass. *From recombination of genes to the estimation of distributions*, volume 4, pages 178–187. Springer, 1996.

Alexander Nareyek. Intelligent agents for computer games. In T.A. Marsland and I. Frank, editors, *Computers and Games, Second International Conference, CG 2002*, pages 414–422, 2002.

Emma Norlig and Liz Sonenberg. An approach to evaluating human characteristics in agents. In Gabriela Lindemann, Daniel Moldt, Mario Paolucci, and Bin Yu, editors, *Proceedings of the International Workshop on Regulated Agent-Based Systems: Theories and Applications (RASTA'02)*, pages 51–60, Bologna, Italy, July 2002.

Jeong Keun Park. Emerging complex behaviors in dynamic multi-agent computer games. M.Sc. thesis, University of Edinburgh, 2003.

Lynee E. Parker. Designing control laws for cooperative agent teams. In *IEEE/ICRA proceedings*, pages 582–587, 1993.

H. V. D. Parunak and S. Bruechner. Ant-line missionaries and cannibals: Synthetic pheromones for distributed motion control. In *Proceedings of the Forth International Conference on Autonomous Agents*, pages 467–474. ACM Press, 2000.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems, Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc, 1988.

Wei Po Lee. *Applying Genetic Programming to Evolve Behavior Primitives and Arbitrators for Mobile Robots*. PhD thesis, University of Edinburgh, 1998.

Marc Ponsen and Pieter Spronck. Improving adaptive game AI with evolutionaey learning. In *Proceedings of the International Conference on Computer Games: Artificial Intelligence, Design and Education*, pages 389–396, Microsoft Campus, Reading, UK, November 2004.

Lisa Rabasca. The internet and computer games reinforce the gender gap. *Monitor On Psychology*, 31(9), October 2000. http://www.apa.org/monitor/oct00/games.html.

Steve Rabin. *AI Game Programming Wisdom*. Charles River Media, Inc, 2002.

Report, Credit Suiss First Boston, May 15 2002.

Craig W. Reynolds. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21(4):25–34, 1987.

Craig W. Reynolds. An evolved, vision-based behavioral model of coordinated group motion. In J. Meyer, H. L. Roitblat, and S. W. Wilson, editors, *From Animals to Animats 2: Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, pages 384–392. The MIT Press/Bradford Books, 1993.

Craig W. Reynolds. Evolution of corridor following behavior in a noisy world. In D. Cliff, P. Husbands, J.-A. Meyer, and S. W. Wilson, editors, *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior SAB-94*, pages 402–410, Cambridge, MA, 1994. MIT Press.

Norman Richards, David E. Moriarty, and Risto Miikulainen. Evolving neural networks to play go. *Appl. Intell.*, 8(1):85–96, 1998.

Justinian Rosca. Generality versus size in genetic programming. In John R. Koza, David E. Goldberg, David B. Fogel, and Rick L. Riolo, editors, *Genetic Programming 1996: Proceedings of the First Annual Conference*, pages 381–387, Stanford University, CA, USA, 1996. MIT Press.

Christopher D. Rosin and Richard K. Belew. Methods for competitive co-evolution: Finding opponents worth beating. In *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 373–380, San Francisco, CA, 1995.

D. Rumelhart, J. McCelland, and the PDP Research Group. *Parallel Distributed Processing*, volume 1. MIT Press, 1986.

D. Rumelhart, G. Hinton, and R. Williams. Learning internal representations by error propagation. In J. Anderson and E. Rosenfeld, editors, *Neurocomputing*, pages 675–695. Cambridge, MA: MIT Press, 1988.

S. Russell and J. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1995.

K. Sims. Evolving virtual creatures. In *Proceedings of the 21th annual Conference on Computer Graphics and Interactive Techniques*, pages 15–22. ACM Press, 1994.

Kenneth Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127, 2002.

Kenneth Stanley, Bobby Bryant, and Risto Miikkulainen. Real-time evolution in the NERO video game. In Graham Kendall and Simon Lucas, editors, *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, pages 182–189, Essex University, Colchester, UK, 4–6 April 2005.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

Penelope Sweetser, Daniel Johnson, Jane Sweetser, and Janet Wiles. Creating engaging artificial characters for games. In *ICEC '03: Proceedings of the second international conference on Entertainment computing*, pages 1–8. Carnegie Mellon University, 2003.

G. Syswerda. Uniform crossover in genetic algorithms. In Schaffer, editor, *Proc. of the Third International Conference on Genetic Algorithms*, pages 2–9, San Mateo, CA, 1989. Morgan Kaufmann.

G. Syswerda. A study of reproduction in generational and steady-state genetic algorithms. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, San Mateo, CA, 1991. Morgan Kaufmann.

N. A. Taatgen, M. van Oploo, J. Braaksma, and J. Niemantsverdriet. How to construct a believable opponent using cognitive modeling in the game of set. In *Proceedings of the fifth international conference on cognitive modeling*, pages 201–206, 2003.

Tim Taylor. Artificial life techniques for generating controllers for physically modelled characters. In Q. Mehdi and N. Gough, editors, *Proceedings of the First International Conference on Intelligent Games and Simulation (GAME-ON 2000)*, 2000.

D. Terzopoulos, X. Tu, and R. Grzeszczuk. Artificial fishes: Autonomous locomotion, perception, behavior, and learning in a simulated physical world. *Artifical Life*, pages 327–351, 1994.

D. Terzopoulos, T. Rabie, and R. Grzeszczuk. Perception and learning in artificial animals. In *Proceedings of the Fifth International Conference on the Synthesis and Simulation of Living Systems*, pages 346–353, 1996.

G. Tesauro. Programming backgammon using self-teaching neural nets. *Artificial Intelligence*, 134:181–199, 2002.

Christian Thurau, Christian Bauckhage, and Gerhard Sagerer. Learning human-like

Movement Behavior for Computer Games. In S. Schaal, A. Ijspeert, A. Billard, Sethu Vijayakumar, J. Hallam, and J.-A. Meyer, editors, *From Animals to Animats 8: Proceedings of the* 8^{th} *International Conference on Simulation of Adaptive Behavior (SAB-04)*, pages 315–323, Santa Monica, LA, CA, July 2004. The MIT Press.

Julian Togelius and Simon Lucas. Forcing neurocontrollers to exploit sensory symmetry through hard-wired modularity in the game of cellz. In Graham Kendall and Simon Lucas, editors, *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, pages 37–43, Essex University, Colchester, UK, 4–6 April 2005.

Ah Chung Tsoi and Andrew D. Back. Locally recurrent globally feedforward networks: A critical review of architectures. *IEEE Transactions on Neural Networks*, 5(2):229–239, March 1994.

Alan M. Turing. Computing machinery and intelligence. *Mind*, 59:433–460, 1950.

Michael van Lent and John E. Laird. Developing an artificial intelligence engine. 1999.

J. Vomlel. Bayesian networks in mastermind. In *Proceedings of the 7th Czech-Japan Seminar*, 2004.

J. Weizenbaum. ELIZA — a computer program for the study of natural language communications between men and machines. *Communications of the Association for Computing Machinery*, 9:36–45, 1966.

G. M. Werner. Evolution of herding behavior in artificial animals. In Meyer, Roitblat, and Wilson, editors, *From Animals to Animats 2: Proceedings of the Second International Conference on Simulation of Adaptive Behavior (SAB-92)*, pages 108–115, Cambridge, MA, 1993. MIT Press.

D. Whitley, T. Starkweather, and C. Bogart. Genetic algorithms and neural networks: Optimizing connections and connectivity. *Parallel Computing*, 14(3):347–361, 1990.

Wikipedia, the Free Encyclodedia. Definition of the word ‘game’, February 2005. http://en.wikipedia.org/wiki/Game.

Steven Woodcock. Game AI: The State of the Industry 2000-2001: It’s not Just Art, It’s Engineering. August 2001.

L. Yaeger. Computational genetics, physiology, metabolism, neural systems, learning, vision, and behaviour of polyworld: Life in a new context. In C. G. Langton, editor, *Artificial Life III: Proceedings of the Workshop on Artificial Life*, volume XVII, pages 263–298, Reading, MA, 1993. Sante Fe Institute Studies in the Sciences and Complexity, Addison-Wesley.

Georgios N. Yannakakis and John Hallam. Evolving Opponents for Interesting Interactive Computer Games. In S. Schaal, A. Ijspeert, A. Billard, Sethu Vijayakumar, J. Hallam, and J.-A. Meyer, editors, *From Animals to Animats 8: Proceedings of the 8^{th} International Conference on Simulation of Adaptive Behavior (SAB-04)*, pages 499–508, Santa Monica, LA, CA, July 2004. The MIT Press.

Georgios N. Yannakakis and John Hallam. Interactive Opponents Generate Interesting Games. In *Proceedings of the International Conference on Computer Games: Artificial Intelligence, Design and Education*, pages 240–247, Microsoft Campus, Reading, UK, November 2004.

Georgios N. Yannakakis and John Hallam. A Generic Approach for Obtaining Higher Entertainment in Predator/Prey Computer Games. *Journal of Game Development*, 2005. in press.

Georgios N. Yannakakis and John Hallam. A generic approach for generating interesting interactive pac-man opponents. In Graham Kendall and Simon Lucas, editors, *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, pages 94–101, Essex University, Colchester, UK, 4–6 April 2005.

Georgios N. Yannakakis and John Hallam. A scheme for creating digital entertainment with substance. In *Proceedings of the Workshop on Reasoning, Representation, and Learning in Computer Games, 19th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 119–124, August 2005.

Georgios N. Yannakakis and John Hallam. Towards Optimizing Entertainment in Computer Games. submitted to Applied Artificial Intelligence, June 2005.

Georgios N. Yannakakis and Manolis Maragoudakis. Player modeling impact on player's entertainment in computer games. In *Proceedings of the 10^{th} International Conference on User Modeling; Lecture Notes in Computer Science*, volume 3538, pages 74–78, Edinburgh, 24–30 July 2005. Springer-Verlag.

Georgios N. Yannakakis, John Levine, John Hallam, and Markos Papageorgiou. Per-

formance, robustness and effort cost comparison of machine learning mechanisms in *FlatLand*. In *Proceedings of the 11th Mediterranean Conference on Control and Automation MED'03*. IEEE, June 2003.

Georgios N. Yannakakis, John Levine, and John Hallam. An Evolutionary Approach for Interactive Computer Games. In *Proceedings of the Congress on Evolutionary Computation (CEC-04)*, pages 986–993, June 2004.

Georgios N. Yannakakis, John Hallam, and John Levine. Evolutionary computation variants for cooperative spatial coordination. In *Congress on Evolutionary Computation (CEC-05)*, pages 2715–2722, Edinburgh, UK, September 2005.

Georgios N. Yannakakis, John Levine, and John Hallam. Emerging Cooperation with Minimal Effort. Rewarding over Mimicking. *IEEE Transactions on Evolutionary Computation*, 2005. accepted.

Georgios N. Yannakakis. Evolutionary computation: Research on emerging strategies through a complex multi-agent environment. Master's thesis, Technical University of Crete, Chania, Greece, September 2001.

X. Yao and Y. Liu. Towards designing artificial neural networks by evolution. *Appl. Math. Computation*, 9(1):54–65, 1996.

X. Yao. Evolutionary artificial neural networks. In A. Kent, J. G. Williams, and C. M. Hall, editors, *Encyclopedia of Computer Science and Technology*, volume 33, pages 137–170. New York: Marcel Dekker Inc., 1995.

X. Yao. Evolving artificial neural networks. In *Proceedings of the IEEE*, volume 87, pages 1423–1447, 1999.

Nahum Zaera, Dave Cliff, and Janet Bruten. (Not) evolving collective behaviors in synthetic fish. In P. Maes, M. Mataric, Jean-Arcady Meyer, J. Pollack, and S. Wilson, editors, *From Animals to Animats 4: Proceedings of the Forth International Conference on Simulation of Adaptive Behavior (SAB'96)*, pages 635–644. The MIT Press/Bradford Books, 1996.